Universidade Federal do Rio de Janeiro
Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia

Departamento de Engenharia Elétrica

COE782 - Introdução ao Aprendizado de Máquina

Prof. Dr. Markus Vinícius Santos Lima

Lista 2 de exercícios

Luiz Henrique Souza Caldas
email: lhscaldas@cos.ufrj.br

4 de junho de 2024

Exercícios do Bishop

1. **Exercício 2.1**

   Verifique que a distribuição de Bernoulli $Bern(x|\mu) = \mu^x(1-\mu)^{1-x}$ satisfaz as seguintes propriedades:

$$\sum_{x=0}^{1} p(x|\mu) = 1$$

$$\mathbb{E}[x] = \mu$$

$$var[x] = \mu(1-\mu).$$

   Mostre que a entropia $H[x]$ de uma variável binária x com distribuição Bernoulli é dada por

$$H[x] = -\mu \ln \mu - (1-\mu)\ln(1-\mu).$$

   Solução:

   $\sum_{x=0}^{1} p(x|\mu) = \mu^1(1-\mu)^{(1-1)} + \mu^0(1-\mu)^{(1-0)} = \mu + (1-\mu) = \underline{1 \quad \rfloor}$

   $\mathbb{E}[x] = \sum_{x=0}^{1} p(x|\mu)x = \sum_{x=0}^{1} \mu^x(1-\mu)^{1-x}x = \mu^1(1-\mu)^{(1-1)}1 + \mu^0(1-\mu)^{(1-0)}0 = \underline{\mu \quad \rfloor}$

   $var[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2 = \sum_{x=0}^{1} \mu^x(1-\mu)^{1-x}x^2 - \mu^2 = \mu - \mu^2 = \underline{\mu(1-\mu) \quad \rfloor}$

   $H[x] = -\sum_x p(x)\ln p(x) = -\sum_x \mu^x(1-\mu)^{1-x} \ln \mu^x(1-\mu)^{1-x}$

   $H[x] = -\mu^1(1-\mu)^{1-1} \ln \mu^1(1-\mu)^{1-1} - \mu^0(1-\mu)^{1-0} \ln \mu^0(1-\mu)^{1-0}$

   $H[x] = \underline{-\mu \ln \mu - (1-\mu)\ln(1-\mu) \quad \rfloor}$

2. **Exercício 2.2**

   A forma da distribuição de Bernoulli dada por $Bern(x|\mu) = \mu^x(1-\mu)^{1-x}$ não é simétrica entre os dois valores de x. Em algumas situações, será mais conveniente usar uma formulação equivalente para a qual ($x \in \{-1, 1\}$), caso em que a distribuição pode ser escrita

$$p(x|\mu) = \left(\frac{1-\mu}{2}\right)^{(1-x)/2} \left(\frac{1+\mu}{2}\right)^{(1+x)/2}$$

   onde $\mu \in [-1, 1]$. Mostre que a distribuiição de $p(x|\mu)$ é normalizada e avalie sua média, variância e entropia.

   Solução:

   $\sum_x p(x|\mu) = \left(\frac{1-\mu}{2}\right)^{(1-(-1))/2} \left(\frac{1+\mu}{2}\right)^{(1+(-1))/2} + \left(\frac{1-\mu}{2}\right)^{(1-(1))/2} \left(\frac{1+\mu}{2}\right)^{(1+(1))/2}$

   $\sum_x p(x|\mu) = \left(\frac{1-\mu}{2}\right) + \left(\frac{1+\mu}{2}\right) = \underline{1 \quad \rfloor}$

   $\mathbb{E}[x] = \sum_{x=0}^{1} p(x|\mu)x = \sum_{x=-1}^{1} \left(\frac{1-\mu}{2}\right)^{(1-x)/2} \left(\frac{1+\mu}{2}\right)^{(1+x)/2} x$

$$\mathbb{E}[x] = \left(\frac{1-\mu}{2}\right)^{(1-(-1))/2} \left(\frac{1+\mu}{2}\right)^{(1+(-1))/2} (-1) + \left(\frac{1-\mu}{2}\right)^{(1-(1))/2} \left(\frac{1+\mu}{2}\right)^{(1+(1))/2} (1)$$

$$\mathbb{E}[x] = \left(\frac{1-\mu}{2}\right)(-1) + \left(\frac{1+\mu}{2}\right)(1)$$

$$\mathbb{E}[x] = -\frac{1-\mu}{2} + \frac{1+\mu}{2} = \underline{\mu \quad \rfloor}$$

$$\mathbb{E}[x^2] = \sum_{x=0}^{1} p(x|\mu)x^2 = \sum_{x=-1}^{1} \left(\frac{1-\mu}{2}\right)^{(1-x)/2} \left(\frac{1+\mu}{2}\right)^{(1+x)/2} x^2$$

$$\mathbb{E}[x^2] = \left(\frac{1-\mu}{2}\right)^{(1-(-1))/2} \left(\frac{1+\mu}{2}\right)^{(1+(-1))/2} (-1)^2 + \left(\frac{1-\mu}{2}\right)^{(1-(1))/2} \left(\frac{1+\mu}{2}\right)^{(1+(1))/2} (1)^2$$

$$\mathbb{E}[x^2] = \left(\frac{1-\mu}{2}\right)(1) + \left(\frac{1+\mu}{2}\right)(1)$$

$$\mathbb{E}[x^2] = \frac{1-\mu}{2} + \frac{1+\mu}{2} = 1$$

$$var[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2 = \underline{1-\mu^2 \quad \rfloor}$$

$$H[x] = -\sum_x p(x) \ln p(x) = \underline{-\left(\frac{1-\mu}{2}\right) \ln \left(\frac{1-\mu}{2}\right) - \left(\frac{1+\mu}{2}\right) \ln \left(\frac{1+\mu}{2}\right) \quad \rfloor}$$

3. **Exercício 2.8**

   Considere duas variáveis x e y com distribuição conjunta $p(x, y)$. Prove o seguinte resultado (resultado bem útil conhecido como Regra da Torre)

   $$\mathbb{E}[x] = \mathbb{E}_y[\mathbb{E}_x[x|y]].$$

   Aqui, $\mathbb{E}_x[x|y]$ denota o valor esperado de x sob a distribuição condicional $p(x|y)$.

   Solução:

   $\mathbb{E}[x] = \int_y \int_x xp(x, y)dydx$

   Mas $p(x, y) = p(x|y)p(y)$ (regra do produto). Então,

   $\mathbb{E}[x] = \int_y \int_x xp(x|y)p(y)dydx = \int_y p(y) \left[\int_x xp(x|y)dx\right] dy$

   $\mathbb{E}[x] = \int_y p(y)\mathbb{E}_x[x|y]dy$

   $\mathbb{E}[x] = \underline{\mathbb{E}_y[\mathbb{E}_x[x|y]] \quad \rfloor}$

4. **Exercício 2.12**

   A distribuição uniforme de uma variável contínua x é definida por

   $$U(x|a, b) = \frac{1}{b-a}, \quad a \leq x \leq b.$$

   Verifique se esta distribuição está normalizada e encontre expressões para sua média e variância.

   Solução:

$$\int_a^b U(x|a,b)dx = \int_a^b \frac{1}{b-a}dx = \frac{1}{b-a}[x]_a^b = \frac{1}{b-a}(b-a) = \underline{1 \quad |}$$

$$\mathbb{E}[x] = \int_a^b \frac{1}{b-a}xdx = \frac{1}{b-a}\left[\frac{x}{2}\right]_a^b = \frac{1}{b-a}\left(\frac{b^2-a^2}{2}\right) = \frac{(b-a)(b+a)}{2(b-a)} = \underline{\frac{a+b}{2} \quad |}$$

$$\mathbb{E}[x^2] = \int_a^b \frac{1}{b-a}x^2dx = \frac{1}{b-a}\left[\frac{x^3}{3}\right]_a^b = \frac{1}{b-a}\left(\frac{b^3-a^3}{2}\right) = \frac{(b-a)(a^2+ab+b^2)}{3(b-a)} = \frac{(a^2+ab+b^2)}{3}$$

$$var[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2 = \frac{(a^2+ab+b^2)}{3} - \frac{(a+b)^2}{4} = \frac{4(a^2+ab+b^2) - 3(a^2+2ab+b^2)}{12}$$

$$var[x] = \frac{a^2-2ab+b^2}{12} = \underline{\frac{(b-a)^2}{12} \quad |}$$

5. **Exercício 2.13**

Avalie a divergência de Kullback-Leibler

$$KL(p||q) = -\int p(x)\ln\left\{\frac{q(x)}{p(x)}\right\}dx$$

entre duas distribuições Gaussianas $p(x) = \mathcal{N}(x|\mu,\Sigma)$ e $q(x) = \mathcal{N}(x|m,L)$ para o caso geral e para os seguintes casos:

(a) ambas as pdfs têm mesma média e matriz de covariância (isto é, p(x) = q(x); caso em que já sabemos quanto a divergência KL deve resultar);

(b) ambas as pds têm a mesma média, isto é, $m = \mu$..

Solução geral:

$$p(x) = \mathcal{N}(x|\mu,\Sigma) = \frac{1}{(2\pi)^{N/2}|\Sigma|^{1/2}}\exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

$$q(x) = \mathcal{N}(x|m,L) = \frac{1}{(2\pi)^{N/2}|L|^{1/2}}\exp\left(-\frac{1}{2}(x-m)^TL^{-1}(x-m)\right)$$

$$KL(p||q) = -\int p(x)\ln\left\{\tfrac{q(x)}{p(x)}\right\}dx$$

$$= -\int p(x)\ln\left\{\tfrac{|\Sigma|^{1/2}}{|L|^{1/2}}\exp\left(-\tfrac{1}{2}(x-m)^TL^{-1}(x-m) + \tfrac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)\right\}dx$$

$$= -\int p(x)\ln\left\{\tfrac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\}dx - \int p(x)\ln\left\{\exp\left(-\tfrac{1}{2}(x-m)^TL^{-1}(x-m) + \tfrac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)\right\}dx$$

$$= -\ln\left\{\tfrac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} - \int p(x)\left(-\tfrac{1}{2}(x-m)^TL^{-1}(x-m) + \tfrac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)dx$$

$$= -\ln\left\{\tfrac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} - \int p(x)\left(-\tfrac{1}{2}(x-m)^TL^{-1}(x-m)\right)dx - \int p(x)\left(\tfrac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)dx$$

$$KL(p||q) = -\ln\left\{\tfrac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} - \mathbb{E}\left[-\tfrac{1}{2}(x-m)^TL^{-1}(x-m)\right] - \mathbb{E}\left[\tfrac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right]$$

$$KL(p||q) = -\ln\left\{\tfrac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} + \tfrac{1}{2}\mathbb{E}\left[(x-m)^TL^{-1}(x-m)\right] - \tfrac{1}{2}\mathbb{E}\left[(x-\mu)^T\Sigma^{-1}(x-\mu)\right]$$

$$KL(p||q) = -\ln\left\{\frac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} + \frac{1}{2}tr\left(L^{-1}\mathbb{E}\left[(x-m)^T(x-m)\right]\right) - \frac{1}{2}tr\left(\Sigma^{-1}\mathbb{E}\left[(x-\mu)^T(x-\mu)\right]\right)$$

$$KL(p||q) = -\ln\left\{\frac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} + \frac{1}{2}tr\left(L^{-1}\mathbb{E}\left[(x-m)^T(x-m)\right]\right) - \frac{1}{2}tr\left(\Sigma^{-1}\mathbb{E}\left[(x-\mu)^T(x-\mu)\right]\right)$$

Como μ é a média de $p(x)$, temos

$$KL(p||q) = -\ln\left\{\frac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} + \frac{1}{2}tr\left(L^{-1}\mathbb{E}\left[(x-m)^T(x-m)\right]\right) - \frac{1}{2}tr\left(\Sigma^{-1}\Sigma\right)$$

$$KL(p||q) = -\ln\left\{\frac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} - \frac{1}{2}tr\left(I\right) + \frac{1}{2}tr\left(L^{-1}\mathbb{E}\left[(x-m)^T(x-m)\right]\right)$$

$$\underline{KL(p||q) = -\ln\left\{\frac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} - \frac{D}{2} + \frac{1}{2}tr\left(L^{-1}\mathbb{E}\left[(x-m)^T(x-m)\right]\right) \quad |}$$

Solução para $\Sigma = L$ e $\mu = m$:

$$KL(p||q) = -\ln\left\{\frac{|\Sigma|^{1/2}}{|\Sigma|^{1/2}}\right\} - \frac{D}{2} + \frac{1}{2}tr\left(\Sigma^{-1}\mathbb{E}\left[(x-\mu)^T(x-\mu)\right]\right)$$

$$KL(p||q) = -\ln\{1\} - \frac{D}{2} + \frac{1}{2}tr\left(\Sigma^{-1}\Sigma\right)$$

$$KL(p||q) = -0 - \frac{D}{2} + \frac{1}{2}tr\left(I\right)$$

$$KL(p||q) = -\frac{D}{2} + \frac{D}{2}$$

$$\underline{KL(p||q) = 0 \quad |}$$

Solução para apenas $\mu = m$:

$$KL(p||q) = -\ln\left\{\frac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} - \frac{D}{2} + \frac{1}{2}tr\left(L^{-1}\mathbb{E}\left[(x-\mu)^T(x-\mu)\right]\right)$$

$$KL(p||q) = -\ln\left\{\frac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} - \frac{D}{2} + \frac{1}{2}tr\left(L^{-1}L\right)$$

$$KL(p||q) = -\ln\left\{\frac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} - \frac{D}{2} + \frac{1}{2}tr\left(I\right)$$

$$KL(p||q) = -\ln\left\{\frac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} - \frac{D}{2} + \frac{D}{2}$$

$$\underline{KL(p||q) = -\ln\left\{\frac{|\Sigma|^{1/2}}{|L|^{1/2}}\right\} \quad |}$$

6. **Exercício 2.15**

Mostre que a entropia da Gaussiana multivariada $\mathcal{N}(x|\mu, \Sigma)$ é dada por

$$H[x] = \frac{1}{2}\ln|\Sigma| + \frac{D}{2}(1 + \ln(2\pi))$$

onde D é a dimensionalidade de x.

Solução:

$H[x] = -\int p(x)\ln p(x)dx$ e $p(x) = \mathcal{N}(x|\mu, \Sigma) = \frac{1}{(2\pi)^{N/2}|\Sigma|^{1/2}}\exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$

Simplificando primeiro o ln

$$\ln p(x) = \ln\left[\frac{1}{(2\pi)^{N/2}|\Sigma|^{1/2}}\exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)\right]$$

$$\ln p(x) = -\frac{D}{2}\ln[2\pi] - \frac{1}{2}\ln|\Sigma| - \frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)$$

Substituindo na fórmula da entropia

$$H[x] = -\int p(x)\left(-\frac{D}{2}\ln[2\pi] - \frac{1}{2}\ln|\Sigma| - \frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)dx$$

$$H[x] = -\int p(x)\left(-\frac{D}{2}\ln[2\pi]\right)dx + \int p(x)\left(-\frac{1}{2}\ln|\Sigma|\right)dx + \int p(x)\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)dx$$

$$H[x] = -\left(-\frac{D}{2}\ln[2\pi]\right) - \left(-\frac{1}{2}\ln|\Sigma|\right) - \int p(x)\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)dx$$

$$H[x] = \frac{D}{2}\ln[2\pi] + \frac{1}{2}\ln|\Sigma| - \int p(x)\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)dx$$

$$H[x] = \frac{D}{2}\ln[2\pi] + \frac{1}{2}\ln|\Sigma| + \frac{1}{2}\mathbb{E}\left[(x-\mu)^T\Sigma^{-1}(x-\mu)\right]$$

$$H[x] = \frac{D}{2}\ln[2\pi] + \frac{1}{2}\ln|\Sigma| + \frac{1}{2}tr\left(\Sigma^{-1}\mathbb{E}\left[(x-\mu)^T(x-\mu)\right]\right)$$

$$H[x] = \frac{D}{2}\ln[2\pi] + \frac{1}{2}\ln|\Sigma| + \frac{1}{2}tr\left(\Sigma^{-1}\Sigma\right) = \frac{D}{2}\ln[2\pi] + \frac{1}{2}\ln|\Sigma| + \frac{1}{2}tr\left(I\right)$$

$$H[x] = \frac{D}{2}\ln[2\pi] + \frac{1}{2}\ln|\Sigma| + \frac{D}{2} = \underline{\frac{1}{2}\ln|\Sigma| + \frac{D}{2}\left(1+\ln[2\pi]\right)}$$

7. **Exercício 2.20**

Uma matriz definida positiva Σ pode ser definida como aquela para a qual o forma quadrática

$$a^T\Sigma a$$

é positiva para qualquer valor real do vetor a. Mostre que uma condição necessária e suficiente para Σ ser definida positiva é que todos os autovalores λ_i de Σ, definidos por $\Sigma u_i = \lambda_i u_i$, sejam positivos.

Solução:

Como u_1,...,u_D constituem uma base para o R^D, podemos escrever

$$a = a_1u_1 + a_2u_2 + ... + a_Du_D$$

onde a_1,...,a_D são coeficientes obtidos projetando-se a em u_1,...,u_D.

Então,

$$a^T\Sigma a = (a_1u_1^T + ... + a_Du_D^T)\Sigma(a_1u_1 + ... + a_Du_D)$$

Aplicando a definição $\Sigma u_i = \lambda_i u_i$, temos

$$a^T\Sigma a = (a_1u_1^T + ... + a_Du_D^T)(a_1\lambda_1u_1 + ... + a_D\lambda_Du_D)$$

Como $u_i^Tu_j = 1$ apenas para $i = j$, sendo 0 caso contrário, temos

$$a^T \Sigma a = (a_1^2 \lambda_1 + ... + a_D^2 \lambda_D)$$

Se os autovalores λ_j são sempre positivos, $a_j^2 \lambda_j \geq 0$ para todo e qualquer j, pois a_j será sempre um número real por ser o coeficiente de uma projeção.

Logo, se todos os autovalores λ_j de Σ são positivos, Σ é positiva definida.

Exercícios Extra

E1 Inferência Bayesiana sequencial

Motivado pela Figura 2.3 do livro, reproduza o experimento da jogada de moeda considerando que foram realizadas 5 jogadas e que a probabilidade de se obter cara é dada por '$\mu = 0,7$'. Plote a distribuição a priori e todas as 5 distribuições a posteriori geradas ao longo do processo iterativo. Considere que a distribuição a priori é uma Beta com parâmetros 'a' e 'b' escolhidos da seguinte forma:

$1^{\underline{o}}$ caso: $a = b = 1$

$2^{\underline{o}}$ caso: $a = b = 2$

Compare os resultados obtidos nos 2 casos.

OBS: Para uma comparação justa entre os 2 casos, primeiro gere os 5 dados (saídas do experimento da moeda, amostrados da Bernoulli definida no enunciado) e depois aplique o aprendizado sequencial para os 2 casos (i.e., para ambas as prioris) usando exatamente os mesmos dados gerados.

Figura 1: Figura 2.3 do Bishop

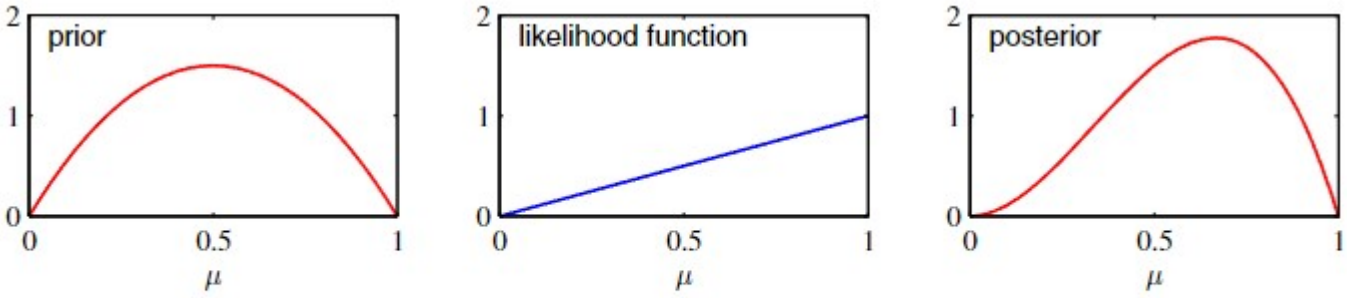


Figure 2.3 Illustration of one step of sequential Bayesian inference. The prior is given by a beta distribution with parameters $a = 2$, $b = 2$, and the likelihood function, given by (2.9) with $N = m = 1$, corresponds to a single observation of $x = 1$, so that the posterior is given by a beta distribution with parameters $a = 3$, $b = 2$.

Solução:

De forma a atender a este exercício foi implementado em python uma função que recebe como entrada uma amostra com 5 dados, a qual é gerada por uma distribuição de Bernoulli com $\mu = 0.7$, e uma lista com dois valores para os parâmetros a e b (um para cada caso pedido). Para cada par (a_0, b_0) a função calcula a priori fazendo Beta$(\mu|a_0, b_0)$ e depois começa a fazer iterações para cada dado da amostra. A cada iteração os valores de a e b são atualizados por

$$a_N = a_0 + m \quad \text{e}$$

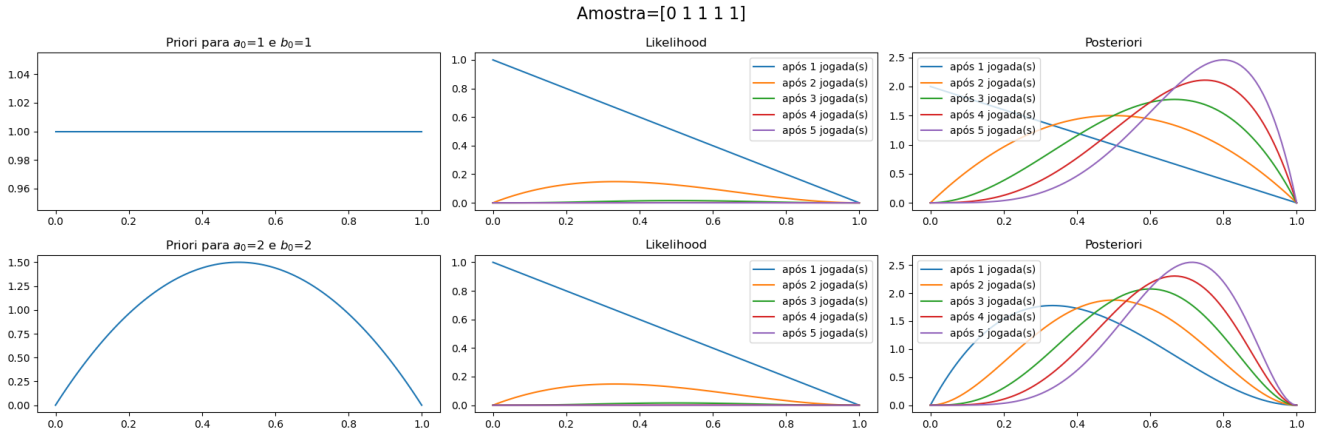
$$b_N = b_0 + (N - m),$$

onde N é o tamanho da amostra até aquela iteração e m é a quantidade de sucessos (caras) até ali. A likelihood é calculada por

$$\prod_{n=1}^{N} \mu^{x_n}(1-\mu)^{(1-x_n)}$$

e a posteriori calcula por Beta$(\mu|a_N, b_N)$. No final são plotados a priori, as 5 likelihoods e as 5 posterioris calculadas. O resultado pode ser observado na figura abaixo.

Figura 2: Plot do resultado



É possível observar como a inferência bayesiana sequencial ajusta a distribuição de probabilidade à medida que novas observações são feitas, atualizando continuamente a distribuição posteriori com base nos novos dados.

A diferença na inicialização da priori com $a = b = 1$ e $a = b = 2$ impacta a sensibilidade da distribuição posteriori aos dados observados. Com $a = b = 1$, a priori é uniforme, implicando que todos os valores de μ são igualmente prováveis inicialmente, tornando a posteriori altamente responsiva aos dados. Já com $a = b = 2$, a priori é mais concentrada em torno de 0.5, refletindo uma crença inicial de que a probabilidade de sucesso é 50% (moeda justa), resultando em uma posteriori mais estável e menos suscetível a variações extremas com poucos dados. Portanto, $a = b = 1$ é adequado quando não há informação prévia, enquanto $a = b = 2$ é útil quando se quer incorporar uma crença prévia moderada.

E2 Verificação experimental do Teorema Central do Limite

Considere a média de N variáveis aleatórias iid. Plote o histograma dessa média considerando que as N variáveis aleatórias têm a seguinte pdf:

1o caso: Uniforme(0,1) - uniforme no intervalo 0 a 1;

2o caso: Bernoulli - escolha o valor do parâmetro como quiser;

Note que, para N suficientemente grande, a distribuição da média converge para uma Gaussiana.

OBS: Usei média ao invés de soma para facilitar a geração do histograma (o eixo horizontal vai ficar fixo, facilitando a comparação para diferentes valores de N, igual na Figura 2.6 do livro).

Figura 3: Figura 2.6 do Bishop

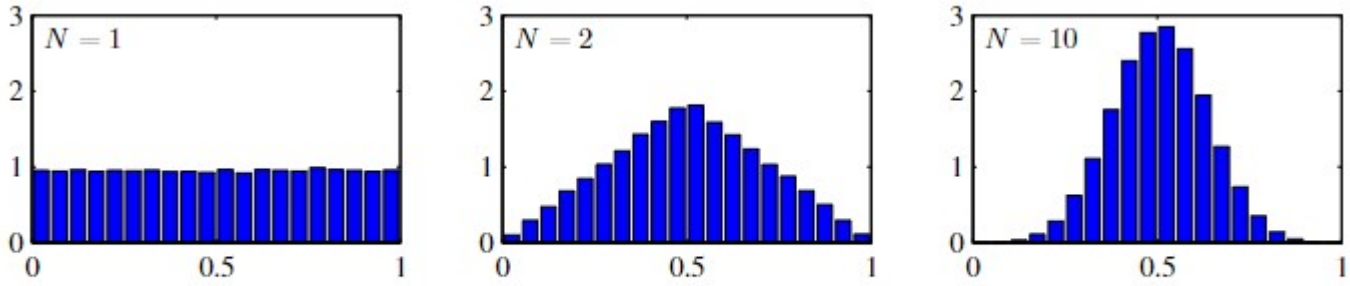
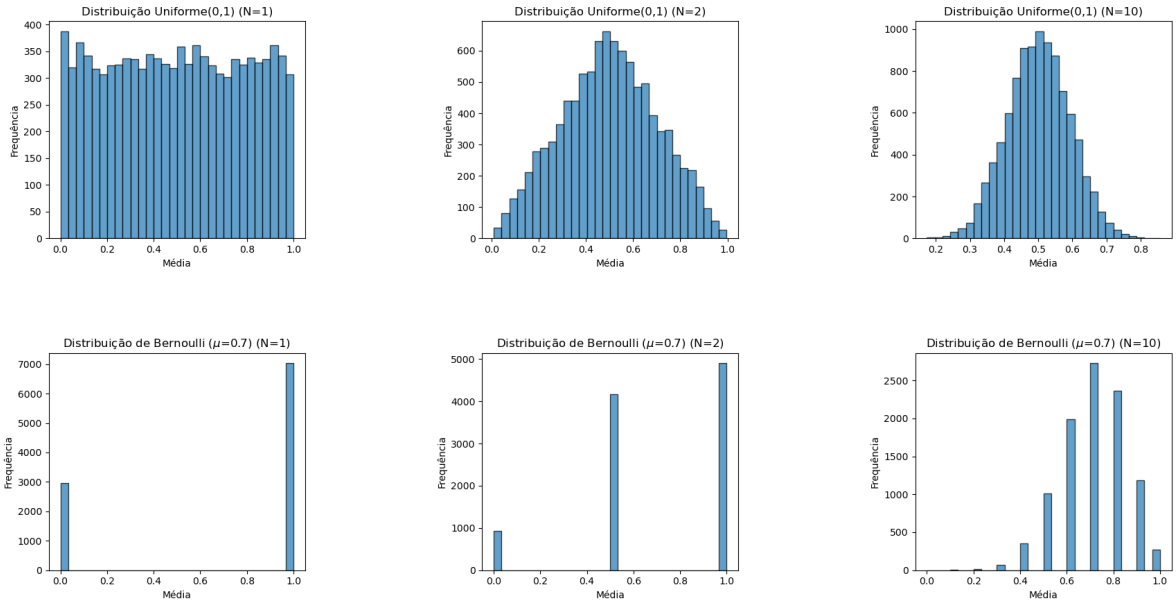


Figure 2.6 Histogram plots of the mean of N uniformly distributed numbers for various values of N. We observe that as N increases, the distribution tends towards a Gaussian.

Solução:

De forma a atender a este exercício foi implementado em python um programa que verifica experimentalmente o Teorema Central do Limite (TCL) ao plotar histogramas das médias de variáveis aleatórias iid (independentemente e identicamente distribuídas) para dois tipos de distribuições: Uniforme(0,1) e Bernoulli(μ). O TCL afirma que, para um número suficientemente grande de variáveis aleatórias iid, a distribuição da média dessas variáveis tende a uma distribuição normal (Gaussiana), independentemente da forma da distribuição original. O código define três valores de N (1, 2 e 10) e gera um número elevado de amostras ($num\_samples = 10000$) para cada valor de N. Para cada conjunto de amostras, calcula-se a média e, em seguida, os histogramas dessas médias são plotados. No caso da distribuição Uniforme$(0, 1)$, amostras são geradas com valores uniformemente distribuídos entre 0 e 1. No caso da distribuição de Bernoulli, amostras são geradas com probabilidade $\mu = 0.7$. Os resultados podem ser observados na figura abaixo.

Figura 4: Plot do resultado



Os resultados mostram que, para N pequeno ($N = 1$ ou $N = 2$), as distribuições das médias ainda refletem a forma original das distribuições de amostra. No entanto, conforme N aumenta para 10, as distribuições das médias começam a se aproximar de uma forma mais simétrica e similar à curva normal, confirmando a previsão do TCL. Este comportamento é mais evidente na distribuição Uniforme(0,1), onde a média das amostras se torna claramente gaussiana. Na distribuição de Bernoulli, o efeito é semelhante, mas a convergência é influenciada pela natureza da distribuição original ($\mu = 0.7$). Essas observações confirmam que, à medida que o tamanho da amostra N aumenta, a distribuição das médias converge para uma distribuição normal, conforme previsto pelo Teorema Central do Limite.

E3 **Verificação experimental da Law of Large Numbers - LLN**

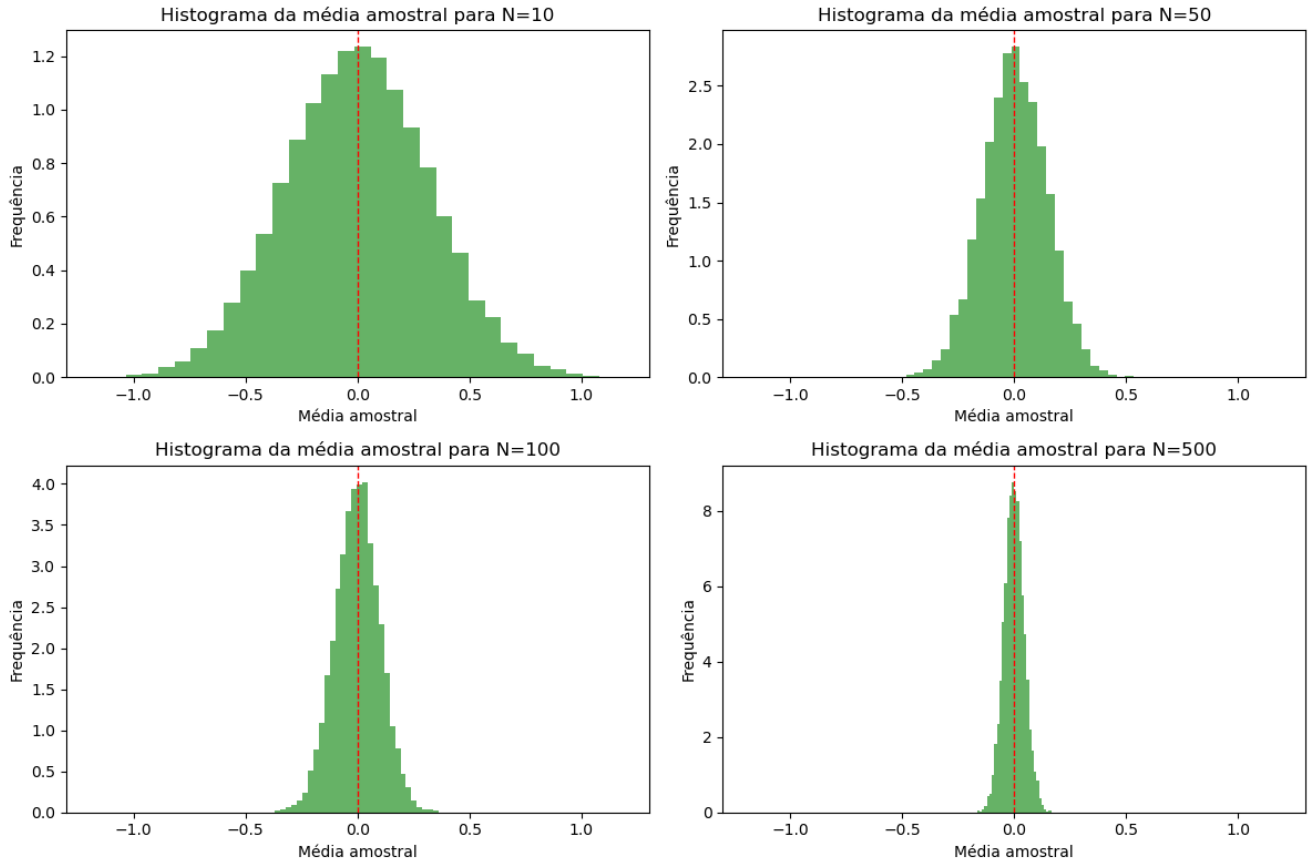
Considere N variáveis aleatórias independentes geradas a partir de uma distribuição normal padrão (isto é, Gaussiana de média 0 e variância 1). Compute o estimador média amostral. Repita o experimento diversas vezes e plote o histograma desse estimador para um dado valor de N. Mostre que, conforme N aumenta, o histograma desse estimador fica cada vez mais estreito em torno do valor correto (i.e., variância vai diminuindo), que é zero.

Solução:

De forma a atender a este exercício foi implementado em python um programa que utiliza o conceito da Lei dos Grandes Números para ilustrar como a média amostral de variáveis aleatórias independentes e identicamente distribuídas (i.i.d.) converge para a média da população conforme o tamanho da amostra (N) aumenta. A Lei dos Grandes Números afirma que, à medida que o número de amostras aumenta, a média amostral se aproxima da média esperada da distribuição. No caso deste código, ele gera N variáveis aleatórias a partir de uma distribuição normal padrão (com média 0 e variância 1), calcula a média dessas variáveis

e repete esse processo 10.000 vezes para diferentes valores de N. Os histogramas das médias amostrais são então plotados para valores crescentes de N, demonstrando que, conforme N aumenta, a variabilidade (ou seja, a largura) das médias amostrais diminui e se concentra em torno de zero, a média real da distribuição. Os resultados podem ser observados na figura abaixo.

Figura 5: Plot do resultado



Pelos resultados, observamos que os histogramas das médias amostrais se tornam progressivamente mais estreitos à medida que o valor de N aumenta. Para $N = 10$, o histograma é relativamente largo, indicando uma maior variação nas médias amostrais. Para $N = 50$ e $N = 100$, o histograma se torna mais concentrado em torno de zero, e para $N = 500$, o histograma é ainda mais estreito. Isso confirma empiricamente a Lei dos Grandes Números, mostrando que a variância da média amostral diminui à medida que o tamanho da amostra aumenta, resultando em uma estimativa mais precisa da média da população. Assim, o experimento visualmente demonstra como a média amostral converge para o valor esperado, reforçando a compreensão teórica desse importante conceito estatístico.

E4 Estimação de pdf

Tente replicar os resultados exibidos nas figuras 2.24 e 2.25 do livro. Para isso, gere uma amostra de N=50 dados cuja distribuição é dada pela curva em verde (corresponde a uma

mistura de 2 gaussianas - veja equação

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k.\Sigma_k)$$

e escolha os parâmetros dessa distribuição da forma que quiser). Estime a pdf do modelo gerador dos dados utilizando 2 métodos: histograma e kernel Gaussiano. Para o parâmetro h, utilize os mesmos valores das figuras.

Figura 6: Figura 2.24 do Bishop

Figure 2.24 An illustration of the histogram approach to density estimation, in which a data set of 50 data points is generated from the distribution shown by the green curve. Histogram density estimates, based on (2.241), with a common bin width Δ are shown for various values of Δ.

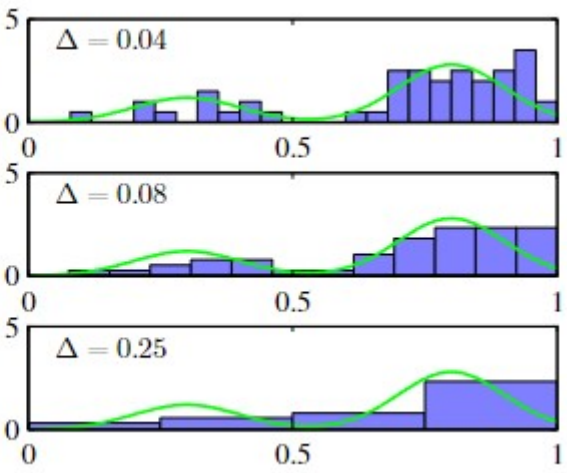
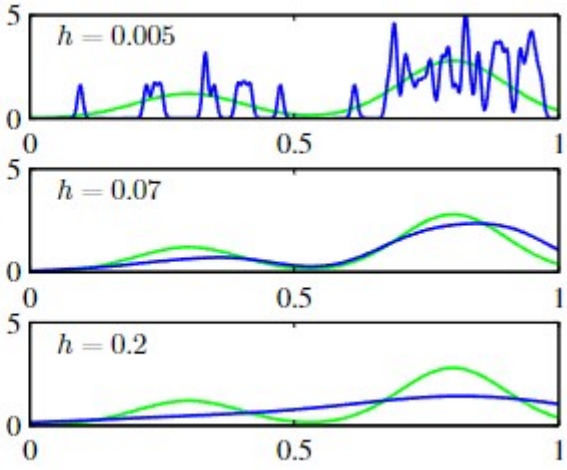


Figura 7: Figura 2.25 do Bishop

Figure 2.25 Illustration of the kernel density model (2.250) applied to the same data set used to demonstrate the histogram approach in Figure 2.24. We see that h acts as a smoothing parameter and that if it is set too small (top panel), the result is a very noisy density model, whereas if it is set too large (bottom panel), then the bimodal nature of the underlying distribution from which the data is generated (shown by the green curve) is washed out. The best density model is obtained for some intermediate value of h (middle panel).



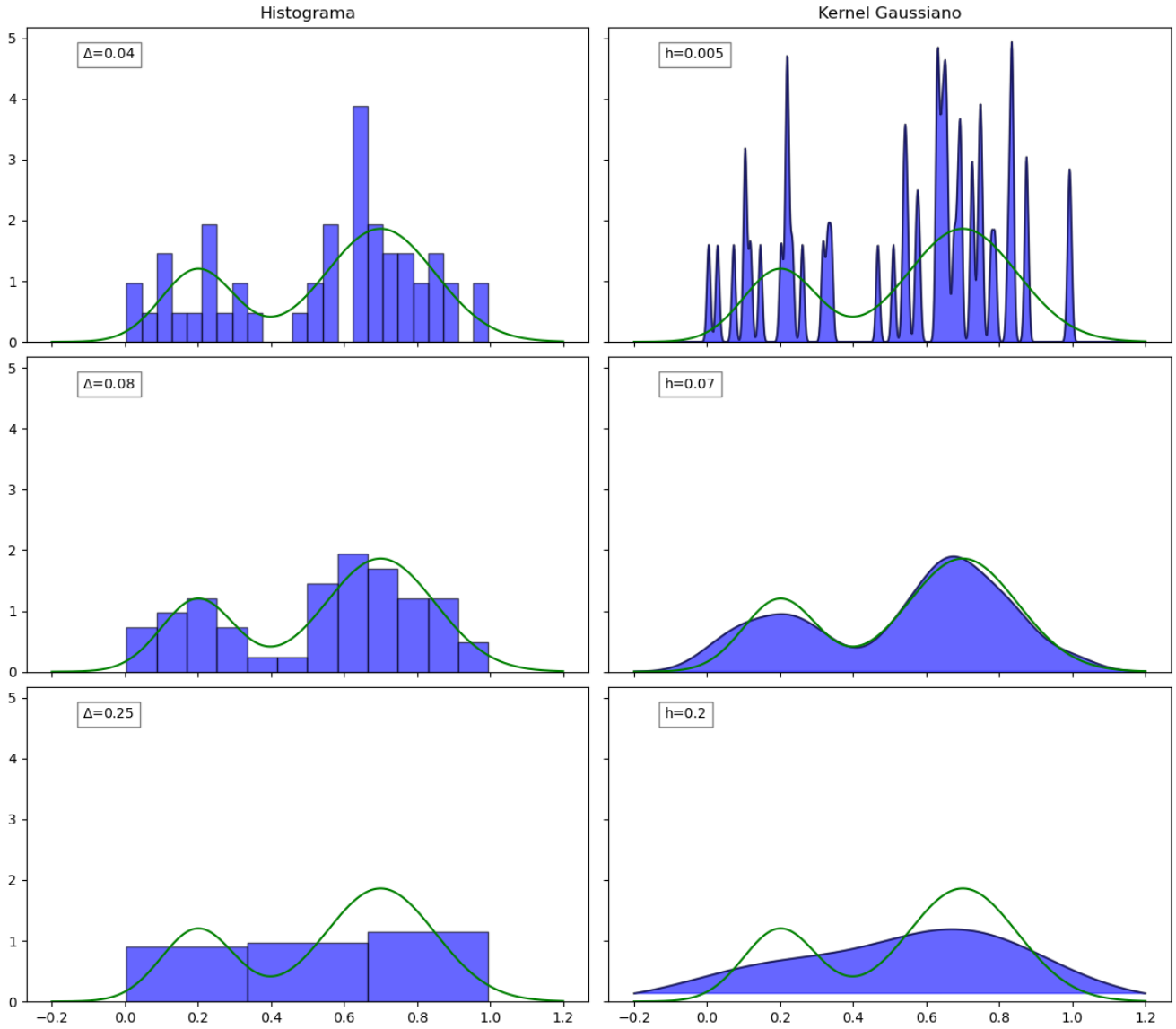
Solução:

O código python implementado resolver esse exercício realiza a estimação da função de densidade de probabilidade (pdf) de um modelo gerador de dados que segue uma distribuição

representada por uma mistura de duas gaussianas. Inicialmente, são definidos os parâmetros das duas distribuições gaussianas, como média (μ), desvio padrão (σ) e peso (π) de cada componente da mistura. Em seguida, é gerada uma amostra de dados com base nessas distribuições.

Para estimar a pdf do modelo gerador, são utilizados dois métodos: histograma e kernel Gaussiano. No método do histograma, a amostra de dados é dividida em intervalos (bins) de largura determinada por diferentes valores de Δ, e a densidade é estimada calculando a frequência relativa de dados em cada bin. No método do kernel Gaussiano, é ajustada uma função de densidade probabilística utilizando a técnica de Kernel Density Estimation (KDE) com diferentes larguras de banda (h). Os resultados podem ser observados na figura abaixo.

Figura 8: Plot do resultado



Na figura, é possível observar os resultados dos dois métodos de estimação de pdf para

diferentes valores de Δ e h. No subplot da esquerda de cada linha, são plotados o histograma da amostra de dados (em azul) e a curva do modelo gerador (em verde). No subplot da direita, são mostradas as curvas de densidade estimadas pelo kernel Gaussiano (em preto, com sombra azul) sobrepostas à curva do modelo gerador (em verde). Além disso, são exibidos os valores de delta e largura de banda utilizados em cada subplot.

Em relação aos resultados, pode-se observar que a estimação da pdf pelo kernel Gaussiano tende a suavizar a distribuição em comparação com o método do histograma, especialmente quando a largura de banda (h) é maior. Isso ocorre porque o kernel Gaussiano utiliza uma abordagem mais suave e contínua para estimar a densidade, enquanto o histograma é mais sensível à escolha dos bins e pode apresentar variações bruscas na representação da distribuição, especialmente com valores pequenos de Δ.

E5 **Classificador K-NN**

Considere 2 classes, C1 e C2, que correspondem aos seguintes modelos geradores:

C1: pdf Gaussiana de média -1 e variância 1

C2: pdf Gaussiana de média 1 e variância 1

Gere 10 observações de cada uma dessas classes e assuma que você sabe exatamente a classe de cada um dos 20 pontos gerados (10 pontos para cada classe). Em seguida, gere mais 2 observações de cada classe e assuma que você NÃO sabe de qual classe esses novos dados pertencem. Utilize a técnica de K-NN, considerando diferentes valores de K, para classificar os 4 novos dados.

OBS: Plote os resultados utilizando cores e símbolos para facilitar a interpretação. Por exemplo: Para os 20 pontos conhecidos, represente-os usando 'bolinhas' vermelhas para C1 e azuis para C2. Para os 4 pontos a serem classificados, mantenha o código de cores (para sabermos identificar qual era a classe correta) e use novos símbolos para identificar se a classificação foi correta (use um 'quadrado') ou se a classificação foi errada (neste caso, use um 'x').

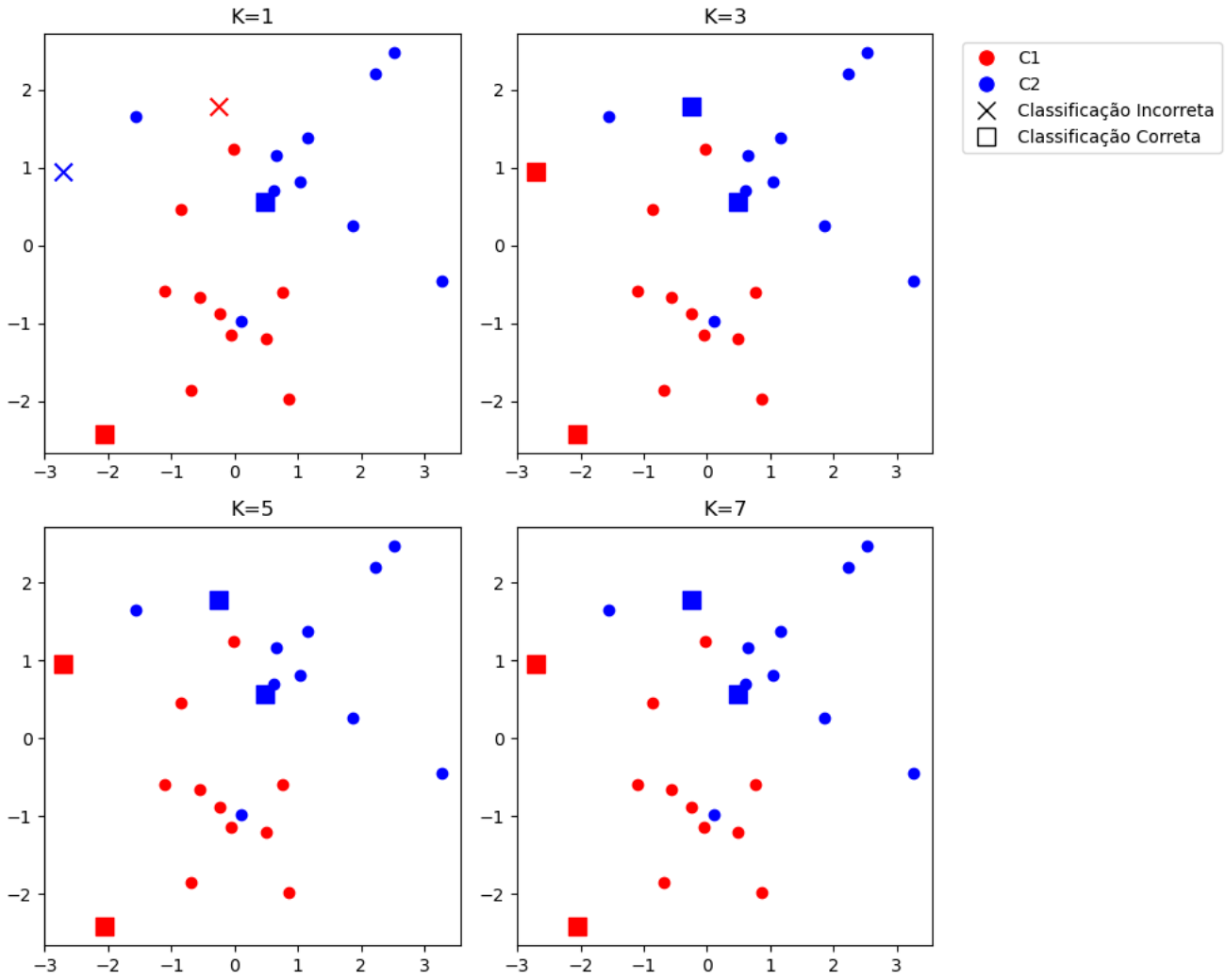
Se for usar outros símbolos e cores, não tem problema. Só não esqueça de fazer uma legenda ou caption que me permita compreender a figura.

Solução:

O código realiza um experimento utilizando o Classificador K-Vizinhos Mais Próximos (K-NN) para classificar dados gerados a partir de duas distribuições Gaussianas diferentes, representando duas classes, C1 e C2. Inicialmente, são gerados 10 pontos para cada classe, conhecidos, e mais 2 pontos de cada classe, inicialmente desconhecidos. O K-NN é aplicado considerando diferentes valores de K (1, 3, 5, 7) para classificar os 4 novos pontos. O K-NN funciona encontrando os K vizinhos mais próximos de um ponto desconhecido e atribuindo a ele a classe mais comum entre esses vizinhos. No código, a classe KNeighborsClassifier é usada para treinar o modelo com os dados conhecidos e prever a classe dos novos pontos. No gráfico resultante, os pontos conhecidos são representados por bolinhas vermelhas (C1) e azuis (C2), enquanto os pontos classificados corretamente recebem um quadrado da mesma cor da classe e os classificados incorretamente recebem um "x"da cor que foram incorretamente classificados. Esse método ilustra como o K-NN pode ser usado para classificar novos

dados com base em dados conhecidos, sendo sensível ao número de vizinhos considerados. Os resultados podem ser observados na figura abaixo.

Figura 9: Plot do resultado



Podemos observar que a classificação dos pontos desconhecidos varia conforme o valor de K. Com K=1, o modelo tende a se ajustar demais aos dados conhecidos, resultando em uma classificação sensível a outliers. Com K=3, 5 e 7, o modelo suaviza essa sensibilidade, e a classificação dos pontos desconhecidos tende a ser mais estável. No entanto, é importante notar que a escolha ideal de K depende do contexto e da natureza dos dados, buscando um equilíbrio entre o viés e a variância do modelo para obter a melhor generalização possível.

Códigos

Código 1: Exercício extra 1

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

def inferencia_bayesiana(amostra, ab_list):

    mu = np.linspace(0, 1, 100)
    n_linhas = len(ab_list)
    fig, ax = plt.subplots(n_linhas, 3, figsize=(18, 6))

    for j in range(n_linhas):
        a0, b0 = ab_list[j]
        # inicialização
        priori = beta.pdf(mu, a0, b0)
        ax[j,0].plot(mu,priori)
        ax[j,0].set_title(f"Priori para $a_0$={a0} e $b_0$={b0}")

        # Atualizações
        for i in range(len(amostra)):
            sub_amostra = amostra[:i+1]
            n_cara = sum(sub_amostra)
            n_coroa = (i+1) - n_cara
            a = a0 + n_cara
            b = b0 + n_coroa
            if i==0:
                likelihood = (mu**n_cara) * ((1 - mu)**n_coroa)
            else:
                likelihood *= (mu**n_cara) * ((1 - mu)**n_coroa)
            posteriori = beta.pdf(mu, a, b)

            # plot
            ax[j,1].plot(mu,likelihood, label=f"após {i+1} jogada(s)")
            ax[j,1].set_title(f"Likelihood")
            ax[j,1].legend()
            ax[j,2].plot(mu,posteriori, label=f"após {i+1} jogada(s)")
            ax[j,2].set_title(f"Posteriori")
            ax[j,2].legend()


    fig.suptitle(f'Amostra={amostra}', fontsize=16)
    plt.tight_layout()
    plt.show()


if __name__ == "__main__":
    # amostra
    mu_true = 0.7
    n = 5
    amostra = bernoulli.rvs(mu_true, size=n)
    ab_list = [(1,1),(2,2)]
```

```
51
52      inferencia_bayesiana(amostra, ab_list)
```

Código 2: Exercício extra 2

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3
4  # Função para plotar histogramas
5  def plot_histogram(means, N, title, ax):
6      ax.hist(means, bins=30, edgecolor='k', alpha=0.7)
7      ax.set_title(f'{title} (N={N})')
8      ax.set_xlabel('Média')
9      ax.set_ylabel('Frequência')
10
11 # Configurações
12 num_samples = 10000  # Número de amostras para calcular a média
13
14 # Casos de N
15 Ns = [1, 2, 10]
16
17
18 fig, ax = plt.subplots(2,3)
19
20 # Caso 1: Variáveis Aleatórias Uniforme(0,1)
21 for i in range(len(Ns)):
22     uniform_samples = np.random.uniform(0, 1, (num_samples, Ns[i]))
23     uniform_means = np.mean(uniform_samples, axis=1)
24     plot_histogram(uniform_means, Ns[i], 'Distribuição Uniforme(0,1)', ax[0,i])
25
26 # Caso 2: Variáveis Aleatórias Bernoulli(p)
27 mu = 0.7  # Parâmetro da distribuição Bernoulli
28 for i in range(len(Ns)):
29     bernoulli_samples = np.random.binomial(1, mu, (num_samples, Ns[i]))
30     bernoulli_means = np.mean(bernoulli_samples, axis=1)
31     plot_histogram(bernoulli_means, Ns[i], f'Distribuição de Bernoulli ($\mu$={
       mu})', ax[1,i])
32
33 plt.tight_layout()
34 plt.show()
```

Código 3: Exercício extra 3

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3
4  # Função para calcular a média amostral de N variáveis aleatórias normais
5  def sample_mean(N, num_experiments):
6      means = np.zeros(num_experiments)
7      for i in range(num_experiments):
8          samples = np.random.normal(0, 1, N)
9          means[i] = np.mean(samples)
10     return means
11
```

```
# Número de experimentos
num_experiments = 10000

# Valores de N para testar
N_values = [10, 50, 100, 500]

# Definindo limites comuns para o eixo x
x_limits = (-1.3, 1.3)

# Plotando os histogramas
fig, axs = plt.subplots(2, 2, figsize=(12, 8))

for i, N in enumerate(N_values):
    row = i // 2
    col = i % 2
    means = sample_mean(N, num_experiments)
    axs[row, col].hist(means, bins=30, density=True, alpha=0.6, color='g')
    axs[row, col].set_title(f'Histograma da média amostral para N={N}')
    axs[row, col].set_xlabel('Média amostral')
    axs[row, col].set_ylabel('Frequência')
    axs[row, col].axvline(0, color='r', linestyle='dashed', linewidth=1)
    axs[row, col].set_xlim(x_limits)  # Ajustando os limites do eixo x


plt.subplots_adjust(hspace=0.25)
plt.tight_layout()
plt.show()
```

Código 4: Exercício extra 4

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.neighbors import KernelDensity

# Parâmetros das distribuições Gaussianas
mu1, sigma1, pi1 = 0.2, 0.1, 0.3
mu2, sigma2, pi2 = 0.7, 0.15, 0.7

# Gerar amostra de dados
N = 50
data = np.hstack([np.random.normal(mu1, sigma1, int(N * pi1)),
                  np.random.normal(mu2, sigma2, int(N * pi2))])

# Valores de delta e largura de banda h
deltas = [0.04, 0.08, 0.25]
bandwidths = [0.005, 0.07, 0.2]

# Plotar
fig, axs = plt.subplots(3, 2, figsize=(12, 16), sharex=True, sharey=True)
x = np.linspace(-0.2, 1.2, 1000)
model_pdf = pi1 * np.exp(-(x - mu1) ** 2 / (2 * sigma1 ** 2)) / (np.sqrt(2 * np
    .pi) * sigma1) \
            + pi2 * np.exp(-(x - mu2) ** 2 / (2 * sigma2 ** 2)) / (np.sqrt(2 *
               np.pi) * sigma2)
```

```
23
24 for i in range(3):
25     axs[i, 0].hist(data, bins=int((max(data) - min(data)) / deltas[i]), density
       =True, alpha=0.6, color='blue', edgecolor='black')
26     axs[i, 0].plot(x, model_pdf, color='green')
27     if i == 0:
28         axs[i, 0].set_title('Histograma')
29     axs[i, 0].text(0.1, 0.9, f'$\Delta$={deltas[i]}', transform=axs[i, 0].
       transAxes, fontsize=10, bbox=dict(facecolor='white', alpha=0.5))
30
31     kde = KernelDensity(kernel='gaussian', bandwidth=bandwidths[i]).fit(data[:,
        np.newaxis])
32     log_dens = kde.score_samples(x[:, np.newaxis])
33     dens = np.exp(log_dens)
34     axs[i, 1].plot(x, dens, alpha=0.6, color='black')
35     axs[i, 1].fill(x, dens, alpha=0.6, color='blue')
36     axs[i, 1].plot(x, model_pdf, color='green')
37     if i == 0:
38         axs[i, 1].set_title('Kernel Gaussiano')
39     axs[i, 1].text(0.1, 0.9, f'h={bandwidths[i]}', transform=axs[i, 1].
       transAxes, fontsize=10, bbox=dict(facecolor='white', alpha=0.5))
40
41 # Ajustar o layout e mostrar a figura
42 plt.tight_layout()
43 plt.show()
```

Código 5: Exercício extra 5

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  from sklearn.neighbors import KNeighborsClassifier
4  from matplotlib.lines import Line2D
5
6  # Gerar dados para C1 e C2
7  np.random.seed(0)
8  C1_data = np.random.normal(loc=-1, scale=1, size=(10, 2))  # 10 pontos para C1
9  C2_data = np.random.normal(loc=1, scale=1, size=(10, 2))   # 10 pontos para C2
10
11 # Gerar novos dados para classificação
12 unknown_data = np.vstack([
13     np.random.normal(loc=-1, scale=1, size=(2, 2)),  # 2 pontos desconhecidos
       para C1
14     np.random.normal(loc=1, scale=1, size=(2, 2))    # 2 pontos desconhecidos
       para C2
15 ])
16
17 # Criar rótulos para os dados conhecidos
18 labels_C1 = np.full(10, 'C1')
19 labels_C2 = np.full(10, 'C2')
20
21 # Criar rótulos verdadeiros para os dados desconhecidos
22 unknown_true_labels = np.array(['C1', 'C1', 'C2', 'C2'])
23
24 # Combinar dados e rótulos para todos os pontos
```

```
X = np.vstack([C1_data, C2_data, unknown_data])
y = np.hstack([labels_C1, labels_C2, unknown_true_labels])

# Configurar o classificador K-NN
k_values = [1, 3, 5, 7]
fig, axs = plt.subplots(2, 2, figsize=(10, 8))

for i, k in enumerate(k_values):
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X[:-4], y[:-4])  # Excluindo os 4 pontos desconhecidos da matriz X e dos rótulos y
    pred = knn.predict(X[-4:])  # Classificar os 4 pontos desconhecidos

    # Plotar os dados conhecidos
    axs[i // 2, i % 2].scatter(C1_data[:, 0], C1_data[:, 1], color='red', label='C1')
    axs[i // 2, i % 2].scatter(C2_data[:, 0], C2_data[:, 1], color='blue', label='C2')

    # Plotar a classificação dos novos dados
    for j, p in enumerate(pred):
        color = 'red' if p == 'C1' else 'blue'
        marker = 's' if p == y[-4:][j] else 'x'  # Marcar corretamente classificados com 's' e incorretamente com 'x'
        axs[i // 2, i % 2].scatter(X[-4:][j, 0], X[-4:][j, 1], color=color, marker=marker, s=100)  # Aumentar o tamanho para melhor visualização
        axs[i // 2, i % 2].set_title(f'K={k}')

legend_elements = [
    Line2D([0], [0], marker='o', color='w', label='C1', markerfacecolor='red', markersize=10),
    Line2D([0], [0], marker='o', color='w', label='C2', markerfacecolor='blue', markersize=10),
    Line2D([0], [0], marker='x', color='w', label='Classificação Incorreta', markeredgecolor='black', markersize=10),
    Line2D([0], [0], marker='s', color='w', label='Classificação Correta', markerfacecolor='white', markeredgecolor='black', markersize=10),
]
axs[0,1].legend(handles=legend_elements, bbox_to_anchor=(1.05, 1.0), loc='upper left')
plt.tight_layout()
plt.show()
```