Lista08_Thiago_Poppe

August 19, 2021

1 Lista 08 - Fundamentos Estatísticos para Ciência dos Dados A

- Aluno: Thiago Martin Poppe
- Matrícula: 2017014324

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
```

2 Exercícios do capítulo 06

2.1 Questão 03)

Consideramos um modelo probabilístico para um problema de diagnóstico de falhas.

Uma variável binária C representa a integridade de uma unidade de disco: $C = 0$ significa que está operando normalmente e $C = 1$ significa que está em estado de falha. Quando a unidade está em funcionamento, monitora-se continuamente usando um temperatura e sensor de choque, e registra duas características binárias, X e Y.

Temos $X = 1$ se o drive foi sujeito a choque (por exemplo, caiu), e $X = 0$, caso contrário. Temos $Y = 1$ se a unidade a temperatura já foi acima de 70^o e $Y = 0$, caso contrário.

A tabela abaixo define a função de massa de probabilidade conjunta dessas três variáveis aleatórias:

[2]:
```
table = pd.DataFrame(
    columns=['x', 'y', 'c', 'prob_conjunta'],
    data=[
        [0, 0, 0, 0.10],
        [0, 1, 0, 0.20],
        [1, 0, 0, 0.20],
        [1, 1, 0, 0.10],
        [0, 0, 1, 0.00],
        [0, 1, 1, 0.10],
        [1, 0, 1, 0.05],
        [1, 1, 1, 0.25]
    ]
)
```

```
table
```

[2]:

	x	y	c	prob_conjunta
0	0	0	0	0.10
1	0	1	0	0.20
2	1	0	0	0.20
3	1	1	0	0.10
4	0	0	1	0.00
5	0	1	1	0.10
6	1	0	1	0.05
7	1	1	1	0.25

- Qual a probabilidade $\mathbb{P}(C = 1)$?

[3]:
```
# Basta fixarmos c = 1 e variar x e y

probs = table[table['c'] == 1]['prob_conjunta']
print('P(C = 1) =', probs.sum())
```

```
P(C = 1) = 0.4
```

- Qual é a probabilidade $\mathbb{P}(C = 0 \,|\, X = 1, Y = 0)$?

[4]:
```
# Basta filtrarmos pela condicional e somarmos as probabilidades onde C = 0
# Note que antes de somarmos nós devemos fazer com que as probabilidades somem 1

aux = table[(table['x'] == 1) & (table['y'] == 0)].copy()
aux['prob_conjunta'] = aux['prob_conjunta'] / aux['prob_conjunta'].sum()

print('P(C = 0 | X = 1, Y = 0) =', aux[aux['c'] == 0]['prob_conjunta'].sum())
```

```
P(C = 0 | X = 1, Y = 0) = 0.8
```

- Qual é a probabilidade $\mathbb{P}(X = 0, Y = 0)$?

[5]:
```
# Basta fixarmos x = 0 e y = 0 e variar o valor de c

probs = table[(table['x'] == 0) & (table['y'] == 0)]['prob_conjunta']
print('P(X = 0, Y = 0) =', probs.sum())
```

```
P(X = 0, Y = 0) = 0.1
```

- Qual é a probabilidade $\mathbb{P}(C = 0|X = 0)$?

[6]:
```
# Basta filtrarmos pela condicional e somarmos as probabilidades onde C = 0
# Note que antes de somarmos nós devemos fazer com que as probabilidades somem 1

aux = table[table['x'] == 0].copy()
```

```
aux['prob_conjunta'] = aux['prob_conjunta'] / aux['prob_conjunta'].sum()

print('P(C = 0 | X = 0) =', aux[aux['c'] == 0]['prob_conjunta'].sum())
```

```
P(C = 0 | X = 0) = 0.75
```

- São X e Y independentes? Justifique sua resposta.

[7]:
```
# Cálculo de P(X = 0, Y = 0)
probs = table[(table['x'] == 0) & (table['y'] == 0)]['prob_conjunta']
print('P(X = 0, Y = 0) =', probs.sum())

# Cálculo de P(X = 0)
probs = table[table['x'] == 0]['prob_conjunta']
print('P(X = 0) =', probs.sum())

# Cálculo de P(Y = 0)
probs = table[table['y'] == 0]['prob_conjunta']
print('P(Y = 0) =', probs.sum())
```

```
P(X = 0, Y = 0) = 0.1
P(X = 0) = 0.4
P(Y = 0) = 0.35000000000000003
```

Resposta: Temos diversas formas de verificar se duas variáveis são independentes. Uma forma simples é verificar se $\mathbb{P}(X = x, Y = y) = \mathbb{P}(X = x) * \mathbb{P}(Y = y)$. Podemos observar que para $x = 0$ e $y = 0$ essa igualdade não é válida, uma vez que teremos que $\mathbb{P}(X = 0, Y = 0) = 0.1$ (como calculado previamente) e que $\mathbb{P}(X = 0) * \mathbb{P}(Y = 0) = 0.4 * 0.35 = 0.14$.

- X e Y são condicionalmente independentes dado C? Isto é, temos $\mathbb{P}(X = x, Y = y \mid C = c) = \mathbb{P}(X = x \mid C = c)\,\mathbb{P}(Y = y \mid C = c)$?

[8]:
```
# Cálculo de P(X = 0, Y = 0 | C = 0)
aux = table[table['c'] == 0].copy()
aux['prob_conjunta'] = aux['prob_conjunta'] / aux['prob_conjunta'].sum()
print('P(X = 0, Y = 0 | C = 0) =', aux[(aux['x'] == 0) & (aux['y'] ==␣
 ↪0)]['prob_conjunta'].sum())

# Cálculo de P(X = 0 | C = 0)
aux = table[table['c'] == 0].copy()
aux['prob_conjunta'] = aux['prob_conjunta'] / aux['prob_conjunta'].sum()
print('P(X = 0 | C = 0) =', aux[aux['x'] == 0]['prob_conjunta'].sum())

# Cálculo de P(Y = 0 | C = 0)
aux = table[table['c'] == 0].copy()
aux['prob_conjunta'] = aux['prob_conjunta'] / aux['prob_conjunta'].sum()
print('P(Y = 0 | C = 0) =', aux[aux['y'] == 0]['prob_conjunta'].sum())
```

```
P(X = 0, Y = 0 | C = 0) = 0.16666666666666669
```

```
P(X = 0 | C = 0) = 0.5
P(Y = 0 | C = 0) = 0.5
```

Resposta: Quando tomamos $x = 0$, $y = 0$ e $c = 0$ percebemos que essa igualdade não é válida. Sendo assim, as variáveis X e Y não são condicionalmente independentes, uma vez que $\mathbb{P}(X = 0, Y = 0 \,|\, C = 0) \approx 0.167$ e que $\mathbb{P}(X = 0 \,|\, C = 0) * \mathbb{P}(Y = 0 \,|\, C = 0) = 0.5 * 0.5 = 0.25$.

3 Exercícios do capítulo 07

3.1 Questão 08)

A distância de Mahalanobis entre um ponto aleatório gaussiano X em $\mathbb{R}^p$ e o seu perfil esperado $\mathbb{E}(X) = \mu$ é dada pela seguinte fórmula, onde Σ é a matriz $p \times p$ de covariância de X:

$$D^2 = (X - \mu)' \, \Sigma^{-1} (X - \mu)$$

- A quantidade D^2 tem um valor típico ou valor esperado $\mathbb{E}(D^2) = ??$
- D^2 possui um afastamento típico de seu valor esperado. Qual o desvio-padrão de D^2, isto é $\sqrt{\mathbb{V}(D^2)} = ??$
- Mais que isto, não somente estes dois resumos da distribuição de D^2 são conhecidos mas a própria distribuição de D^2 é conhecida. $D^2 \sim ??$

Resposta: Quando $X \sim N_p(\mu, \Sigma)$ nós teremos que $D^2 \sim \chi^2_p$, onde χ^2_p denota a distribuição qui-quadrado com p graus de liberdade. Sendo assim, teremos que $\mathbb{E}(D^2) = \mathbb{E}(\chi^2_p) = p$ e que $\sqrt{\mathbb{V}(D^2)} = \sqrt{\mathbb{V}(\chi^2_p)} = \sqrt{2p}$.

- Fixando uma constante c qualquer, o conjunto de pontos $\mathbf{x} \in \mathbb{R}^p$ que satisfazem $D^2 = c$ formam um elipsóide em p dimensões. Isto é, os pontos $\mathbf{x}$ que estão a uma distância D^2 igual a c do seu perfil esperado formam um elipsóide. Quais são os eixos deste elipsóide e os seus tamanhos relativos?

Resposta: Teremos que os eixos deste elipsóide serão iguais aos autovetores da matriz de covariância Σ e os seus tamanhos relativos serão a raiz quadrada dos seus respectivos autovalores.

- E possível mostrar que, com probabilidade $1 - \alpha$, o vetor aleatório X deve cair dentro da elipse $D^2 = c$ onde $c = \chi^2_p(\alpha)$ é o quantil $(1 - \alpha)100\%$ de uma distribuição qui-quadrado com p graus de liberdade onde p é a dimensão do vetor X. No caso particular de um vetor bidimensional, o valor de c associado com a probabilidade $1 - \alpha = 0.95$ é igual a $c = 9.21$. Assim, se $X = (X1, X2)$ estiver fora dessa elipse (isto é, se $D^2 > 9.21$), o ponto pode ser consirado um tanto anômalo ou extremo.

- O arquivo `stiffness.txt` contém quatro tipos de medições da rigidez de pranchas de madeira. A primeira é obtida aplicando-se uma onda de choque através da prancha, a segunda aplicando-se uma vibração à prancha e as outas duas são obtidas por meio de testes estáticos. Assuma que cada uma das 4 medições em uma prancha são instâncias de um vetor $N_4(\mu, \Sigma)$. Estime o vetor μ e a matriz (4×4) Σ usando os dados da amostra. A seguir, usando estes valores estimados como se fossem os verdadeiros valores de μ e Σ, calcule o valor de D^2 para cada ponto da amostra. Quais pontos parecem extremos? Olhando as variáveis INDIVIDUALMENTE ou em pares através de scatterplots seria possível detectar estes pontos extremos? Faça um scatterplot dos dados para entender sua resposta.

[9]:
```
filepath = 'stiffness.txt' # Insira aqui o caminho para os dados
data = pd.read_table('stiffness.txt', sep='\s+', header=None)

# A última coluna do arquivo foi desconsiderada, assumindo apenas as 4 primeiras␣
 ↪como as variáveis
data = data.iloc[:, :4]
data.columns = ['m1', 'm2', 'm3', 'm4']

data.head()
```

[9]:

	m1	m2	m3	m4
0	1889	1651	1561	1778
1	2403	2048	2087	2197
2	2119	1700	1815	2222
3	1645	1627	1110	1533
4	1976	1916	1614	1883

[10]:
```
# Estimando mu e sigma a partir da amostra (arredondaremos os valores para 5␣
 ↪casas decimais)

mu = data.mean()
sigma = np.cov(data.T)

print('Vetor mu calculado:')
print(mu)

print('\nMatriz de covariância encontrada:')
print(sigma)
```

```
Vetor mu calculado:
m1    1906.100000
m2    1749.533333
m3    1509.133333
m4    1724.966667
dtype: float64

Matriz de covariância encontrada:
[[105616.3         94613.53103448  87289.71034483  94230.72758621]
 [ 94613.53103448 101510.11954023  76137.09885057  81064.36321839]
 [ 87289.71034483  76137.09885057  91917.08505747  90352.38390805]
 [ 94230.72758621  81064.36321839  90352.38390805 104227.96436782]]
```

[11]:
```
# Calculando a distância de Mahalanobis para cada ponto da amostra
def mahalanobis_distance(X):
    return (X - mu).T @ np.linalg.inv(sigma) @ (X - mu)

data['D2'] = data.apply(mahalanobis_distance, axis=1)
```

```
data.head()
```

[11]:

	m1	m2	m3	m4	D2
0	1889	1651	1561	1778	0.600013
1	2403	2048	2087	2197	5.477020
2	2119	1700	1815	2222	7.616644
3	1645	1627	1110	1533	5.207610
4	1976	1916	1614	1883	1.398078

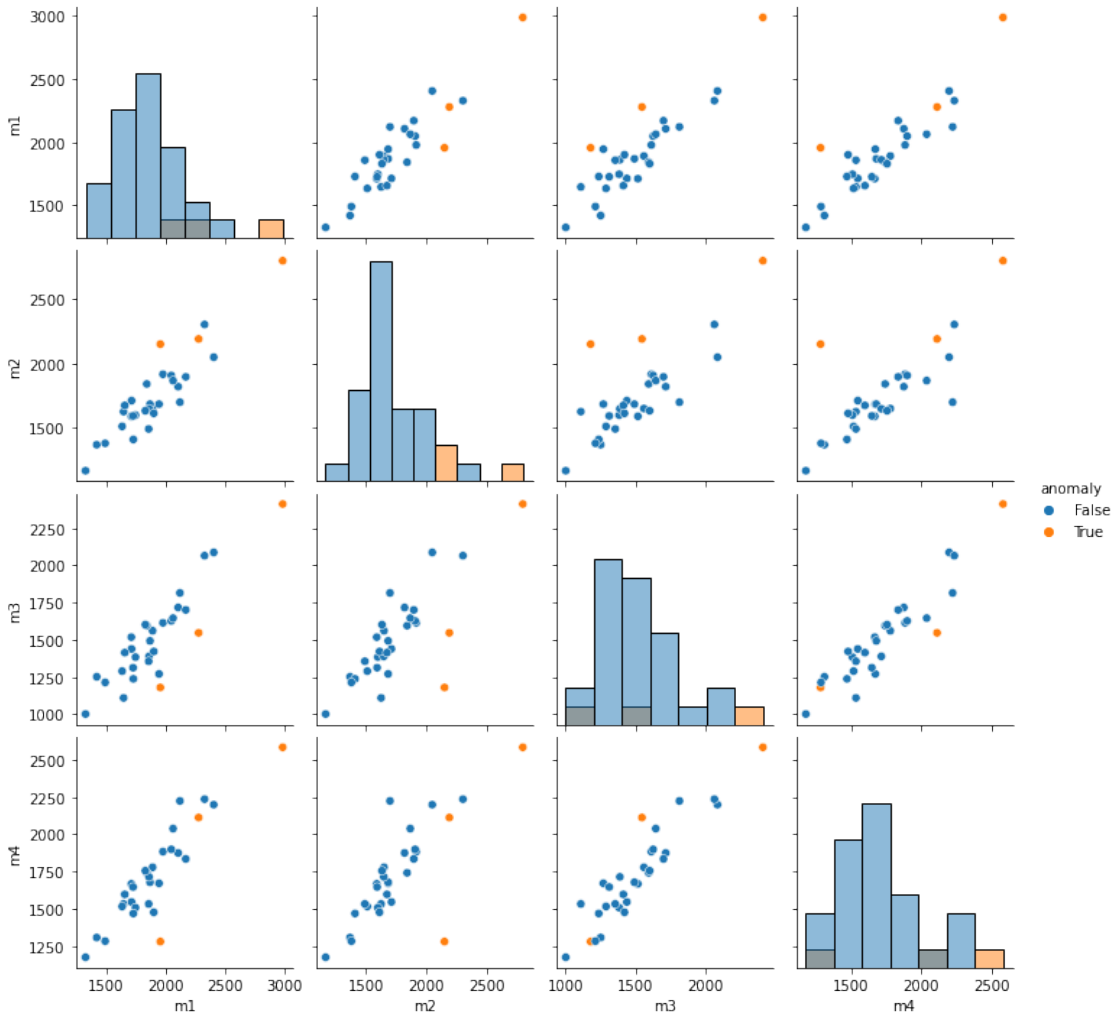
[12]:
```
# Recuperando pontos que são anômalos (D2 > 9.21)
data[data['D2'] > 9.21]
```

[12]:

	m1	m2	m3	m4	D2
8	2983	2794	2412	2581	12.264755
15	1954	2149	1180	1281	16.847407
20	2276	2189	1547	2111	9.898038

[13]:
```
# Criando nova coluna para dizer se um ponto é anômalo ou não (melhor␣
 ↪visualização)
data['anomaly'] = data['D2'].apply(lambda x: x > 9.21)

sns.pairplot(data, vars=data.columns[:4], hue='anomaly', diag_kind='hist')
```

[13]:
```
<seaborn.axisgrid.PairGrid at 0x7fe1f29c0d30>
```



Podemos notar que de fato algumas anomalias podem ser percebidas na análise dos scatterplots (não tão facilmente como observar o valor de D^2), porém, não conseguimos dizer a priori se todas as anomalias poderão ser percebidas através desse método.

3.2 Questão 09)

- **Nota:** Para esse exercício foi necessário o uso do dataset Iris, disponibilizado no `UCI - Machine Learning Repository` em https://archive.ics.uci.edu/ml/datasets/iris.

[14]:
```
filepath = 'iris.data' # Insira aqui o caminho para os dados
iris = pd.read_csv('iris.data', header=None)
iris.columns = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width',␣
 ↪'class']

iris.head()
```

[14]:

	sepal_length	sepal_width	petal_length	petal_width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

- Forneça uma estimativa para o vetor μ, para a matriz Σ e ρ para a classe setosa.

[15]:
```
setosa = iris[iris['class'] == 'Iris-setosa']
setosa = setosa.iloc[:, :4]

mu = np.mean(setosa.values, axis=0)
sigma = np.cov(setosa.T)
rho = np.corrcoef(setosa.T)

print('Estimativa para o vetor mu:')
print(mu)

print('\nEstimativa para a matriz sigma:')
print(sigma)

print('\nEstimativa para rho:')
print(rho)
```

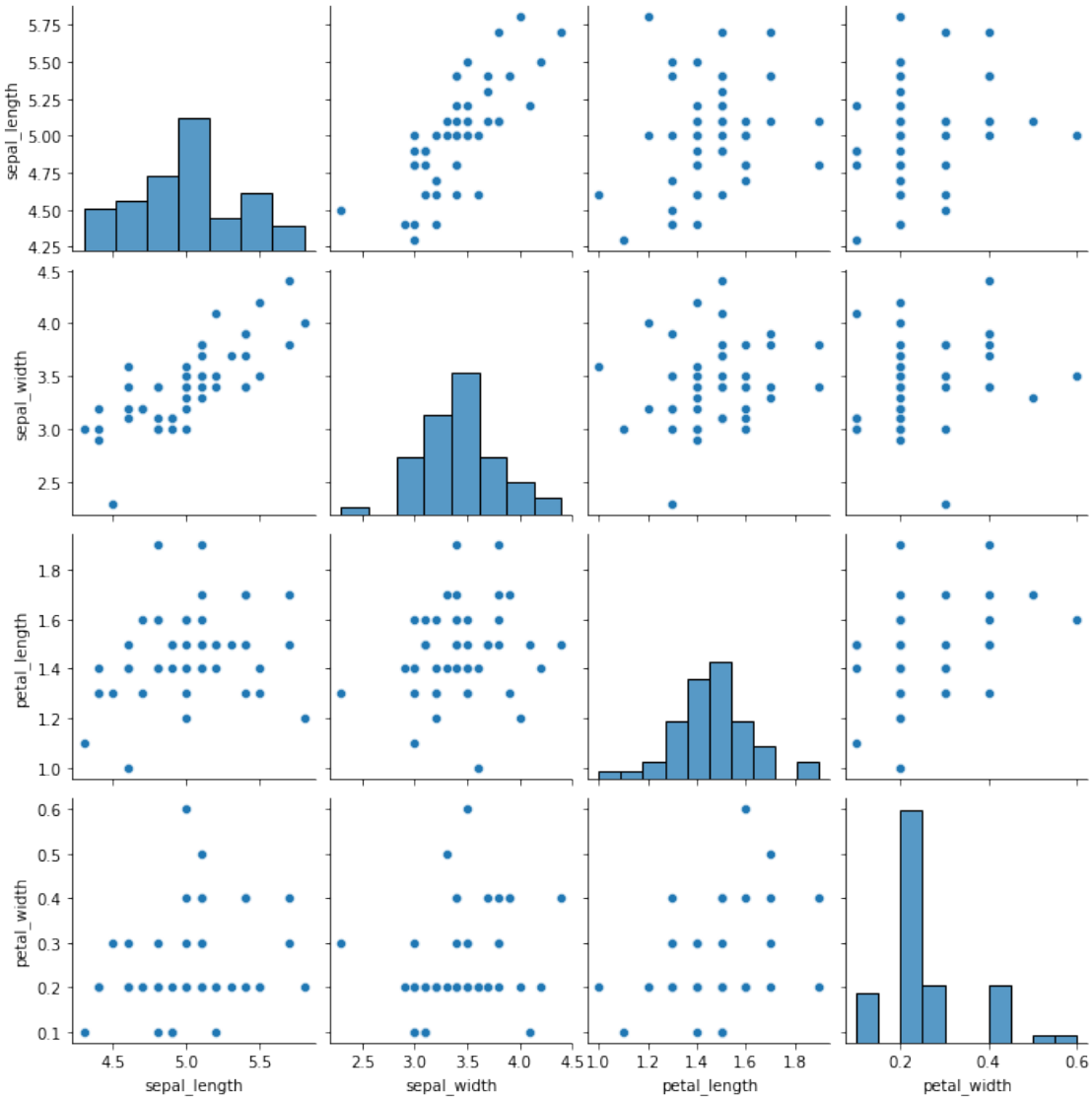
```
Estimativa para o vetor mu:
[5.006 3.418 1.464 0.244]

Estimativa para a matriz sigma:
[[0.12424898 0.10029796 0.01613878 0.01054694]
 [0.10029796 0.14517959 0.01168163 0.01143673]
 [0.01613878 0.01168163 0.03010612 0.00569796]
 [0.01054694 0.01143673 0.00569796 0.01149388]]

Estimativa para rho:
[[1.         0.74678037 0.26387409 0.27909157]
 [0.74678037 1.         0.17669463 0.27997289]
 [0.26387409 0.17669463 1.         0.30630821]
 [0.27909157 0.27997289 0.30630821 1.        ]]
```

- A partir da matriz de correlação entre os pares de v.a.'s (e do plot de dispersão dos pontos), quais as variáveis que são mais correlacionadas? E quais são menos correlacionadas?

Resposta: A partir do plot de dispersão dos pontos abaixoe da matriz de correlação acima, nós percebemos que as variáveis mais correlacionadas são `sepal_width` e `sepal_length`. Já as menos correlacionadas são `sepal_width` e `petal_length`.

[16]:
```
sns.pairplot(setosa, diag_kind='hist')
```

[16]: <seaborn.axisgrid.PairGrid at 0x7fe1e9e267f0>



- Obtenha a distribuição MARGINAL do sub-vetor $X^* = (X_1, X_2)$, o comprimento e largura da sépala.

Resposta: Já que o vetor X segue uma normal multivariada, teremos que X^* também seguirá uma distribuição normal multivariada, com $\bar{}^* = [5.006, 3.418]$ e matriz de covariância igual à (submatriz de Σ):

$$\Sigma^* = \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_2\sigma_1 & \sigma_2^2 \end{pmatrix} = \begin{pmatrix} 0.124 & 0.100 \\ 0.100 & 0.145 \end{pmatrix}$$

- Obtenha a distribuição CONDICIONAL do sub-vetor $X^* = (X_1, X_2)$ quando são conhecidos os valores x_3 e x_4 das v.a.'s (X_3, X_4). Obtenha esta distribuição para dois valores genéricos

x_3 e x_4. A seguir use dois valores específicos: $x_3 = 1.8$ e $x_4 = 0.6$, dois valores relativamente altos para estas variáveis. Compare $DP_1 = \sqrt{\mathbb{V}(X_1)}$ com $\sqrt{\mathbb{V}(X_1 \mid X_3 = 1.8, X_4 = 0.6)}$, o desvio padrão da variável X_1 condicionada nos valores de X_3 e X_4.

[17]:
```
x3, x4 = 1.8, 0.6

sig11 = np.array([
    [0.12424898, 0.10029796],
    [0.10029796, 0.14517959]
])

sig12 = np.array([
    [0.01613878, 0.01054694],
    [0.01168163, 0.01143673]
])

sig21 = np.array([
    [0.01613878, 0.01168163],
    [0.01054694, 0.01143673]
])

sig22 = np.array([
    [0.03010612, 0.00569796],
    [0.00569796, 0.01149388]
])
```

Resposta: Teremos que o sub-vetor X^* seguirá uma distribuição normal multivariada $N_2(\mathbf{m}, \mathbf{V})$ com $\mathbf{m}$ e $\mathbf{V}$ igual à:

$$\begin{aligned}\mathbf{m} &= \begin{pmatrix}\mu_1\\ \mu_2\end{pmatrix} + \Sigma_{12}\Sigma_{22}^{-1}\begin{pmatrix}x_3 - \mu_3\\ x_4 - \mu_4\end{pmatrix}\\ &= \ldots \text{ (cálculos em Python logo abaixo)}\\ &= \begin{pmatrix}5.396\\ 3.807\end{pmatrix}\end{aligned}$$

[18]:
```
mu[[0,1]] + sig12 @ np.linalg.inv(sig22) @ (np.array([x3,x4]) - mu[[2,3]])
```

[18]:
```
array([5.39646372, 3.80738264])
```

$$\begin{aligned}\mathbf{V} &= \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}\\ &= \ldots \text{ (cálculos em Python logo abaixo)}\\ &= \begin{pmatrix}0.110 & 0.087\\ 0.087 & 0.132\end{pmatrix}\end{aligned}$$

[19]:
```
sig11 - sig12 @ np.linalg.inv(sig22) @ sig21
```

[19]:
```
array([[0.11020779, 0.08739917],
       [0.08739917, 0.13247486]])
```

Com isso, teremos que $\sqrt{\mathbb{V}(X_1)} \approx 0.35249$ enquanto que $\sqrt{\mathbb{V}(X_1 \mid X_3 = 1.8, X_4 = 0.6)} \approx 0.33197$.

- Obtenha agora a distribuição CONDICIONAL do sub-vetor $X^* = (X_1, X_2)$ quando é conhecido apenas o valor de X_3.

Resposta: Podemos raciocinar de forma similar à questão anterior, porém dessa vez será como se a variável X_4 não existisse. Portanto, trabalharemos apenas com a seguinte matriz:

$$\Sigma = \begin{pmatrix} 0.124 & 0.1 & 0.016 \\ 0.1 & 0.145 & 0.012 \\ 0.016 & 0.012 & 0.03 \end{pmatrix}$$

Da mesma forma, iremos calcular os valores de **m** e **V** da seguinte forma:

$$\begin{aligned} \mathbf{m} &= \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} + \Sigma_{12}\Sigma_{22}^{-1}\left(x_3 - \mu_3\right) \\ &= \begin{pmatrix} 5.006 \\ 3.418 \end{pmatrix} + \begin{pmatrix} 0.016 \\ 0.012 \end{pmatrix} (0.03)^{-1} \left(x_3 - 1.464\right) \\ &= \ldots \text{ (cálculos em Python logo abaixo)} \\ &= \begin{pmatrix} 5.185 \\ 3.552 \end{pmatrix} \end{aligned}$$

[20]:
```
m0 = np.array([[5.006], [3.418]])
m1 = np.array([[0.016], [0.012]])
m2 = np.array([[0.03]])
m3 = np.array([[x3 - 1.464]])

m0 + m1 @ np.linalg.inv(m2) @ m3
```

[20]:
```
array([[5.1852],
       [3.5524]])
```

$$\begin{aligned} \mathbf{V} &= \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \\ &= \begin{pmatrix} 0.124 & 0.1 \\ 0.1 & 0.145 \end{pmatrix} - \begin{pmatrix} 0.016 \\ 0.012 \end{pmatrix} (0.03)^{-1} \begin{pmatrix} 0.016 & 0.012 \end{pmatrix} \\ &= \ldots \text{ (cálculos em Python logo abaixo)} \\ &= \begin{pmatrix} 0.115 & 0.094 \\ 0.094 & 0.140 \end{pmatrix} \end{aligned}$$

[21]:
```
m0 = np.array([[0.124, 0.1], [0.1, 0.145]])
m1 = np.array([[0.016], [0.012]])
m2 = np.array([[0.03]])
m3 = np.array([[0.016, 0.012]])
```

```
m0 - m1 @ np.linalg.inv(m2) @ m3
```

[21]:
```
array([[0.11546667, 0.0936    ],
       [0.0936    , 0.1402    ]])
```

- Obtenha também distribuição CONDICIONAL do sub-vetor $X^* = (X_1, X_2)$ quando é conhecido apenas o valor de X_4.

Resposta: Teremos o **mesmo** procedimento da questão anterior, porém, dessa vez excluíndo a variável X_3.

$$\Sigma = \begin{pmatrix} 0.124 & 0.1 & 0.011 \\ 0.1 & 0.145 & 0.011 \\ 0.011 & 0.011 & 0.011 \end{pmatrix}$$

Da mesma forma, iremos calcular os valores de **m** e **V** da seguinte forma:

$$\begin{aligned} \mathbf{m} &= \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} + \Sigma_{12}\Sigma_{22}^{-1}\left(x_4 - \mu_4\right) \\ &= \begin{pmatrix} 5.006 \\ 3.418 \end{pmatrix} + \begin{pmatrix} 0.011 \\ 0.011 \end{pmatrix}\left(0.011\right)^{-1}\left(x_4 - 0.244\right) \\ &= \ldots \text{ (cálculos em Python logo abaixo)} \\ &= \begin{pmatrix} 5.362 \\ 3.774 \end{pmatrix} \end{aligned}$$

[22]:
```
m0 = np.array([[5.006], [3.418]])
m1 = np.array([[0.011], [0.011]])
m2 = np.array([[0.011]])
m3 = np.array([[x4 - 0.244]])

m0 + m1 @ np.linalg.inv(m2) @ m3
```

[22]:
```
array([[5.362],
       [3.774]])
```

$$\begin{aligned} \mathbf{V} &= \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \\ &= \begin{pmatrix} 0.124 & 0.1 \\ 0.1 & 0.145 \end{pmatrix} - \begin{pmatrix} 0.011 \\ 0.011 \end{pmatrix}\left(0.011\right)^{-1}\begin{pmatrix} 0.011 & 0.011 \end{pmatrix} \\ &= \ldots \text{ (cálculos em Python logo abaixo)} \\ &= \begin{pmatrix} 0.113 & 0.089 \\ 0.089 & 0.134 \end{pmatrix} \end{aligned}$$

[23]:
```
m0 = np.array([[0.124, 0.1], [0.1, 0.145]])
m1 = np.array([[0.011], [0.011]])
m2 = np.array([[0.011]])
m3 = np.array([[0.011, 0.011]])

m0 - m1 @ np.linalg.inv(m2) @ m3
```

[23]:
```
array([[0.113, 0.089],
       [0.089, 0.134]])
```

- Comparando as três últimas respostas que você forneceu, qual das duas variáveis isoladamente, X_3 ou X_4, diminui a incerteza acerca de X_2 mais fortemente? Isto é, se você tivesse de escolher apenas uma delas, X_3 ou X_4, qual você iria preferir se seu objetivo fosse predizer o valor de X_2?

Resposta:: A variável X_4 diminui a incerteza mais fortemente, uma vez que $\mathbb{V}(X_2 \mid X_4) = 0.134$ é menor que $\mathbb{V}(X_2 \mid X_3) = 0.140$.

- Considere a melhor preditora para X_2 que você escolheu, dentre X_3 ou X_4, na questão anterior. Digamos que tenha sido X_4. Avalie quanto conhecer a outra variável (neste caso, X_3) reduz ADICIONALMENTE a incerteza acerca de X_3. Isto é, compare $\mathbb{V}(X_2 \mid X_4)$ com $\mathbb{V}(X_2 \mid X_3, X_4)$.

Resposta: A variável escolhida foi X_4. Teremos que $\mathbb{V}(X_2 \mid X_3, X_4) = 0.132$, sendo ainda menor do que conhecer apenas X_4, mas devemos estar cientes de possíveis erros de arredondamento. Além disso, em uma análise anterior, concluímos que a correlação entre X_2 e X_3 é a menor de todas, possivelmente não influenciando tanto assim na predição de X_2.

3.3 Questão 10)

- Seja $\mathbf{X} = (X_1, X_2, X_3)'$ um vetor aleatório com distribuição normal multivariada com $\mu = (-1, 0, 2)'$ e

$$\Sigma = \begin{pmatrix} 1 & -2 & 0 \\ -2 & 5 & 0 \\ 0 & 0 & 2 \end{pmatrix}$$

- Seja $\mathbf{b}$ um vetor k-dimensional e $\mathbf{C}$ uma matriz $k \times 3$ formada por constantes. Uma das propriedades da normal multivariada é que a distribuição do vetor $\mathbf{b} + \mathbf{CX}$ de dimensão k é normal com vetor de médias $\mathbf{b} + \mathbf{C}\mu$ e matriz de $k \times k$ covariância $\mathbf{C\Sigma C}^T$. Use esta propriedade para obter a distribuição das seguintes variáveis:

- Distribuição marginal de X_1, de X_2 e de X_3.

[24]:
```
mu = np.array([[-1, 0, 2]]).T

sigma = np.array([
    [1, -2, 0],
    [-2, 5, 0],
    [0, 0, 2]
])
```

Resposta: $X_1 \sim N(-1,1)$, $X_2 \sim N(0,5)$ e $X_3 \sim N(2,2)$.

- Distribuição de um indicador composto pelas 3 variáveis: $T = 0.4X_1 + 0.3X_2 + 0.3X_3$.

Resposta: Teremos que $\mathbf{C} = (0.4, 0.3, 0.3)'$ com $\mathbf{b} = \mathbf{0}$ e que $T \sim N_{k=1}(\mathbf{C}\mu, \mathbf{C}\Sigma\mathbf{C}^{\mathbf{T}})$, onde $\mathbf{C}\mu = 0.4\mu_1 + 0.3\mu_2 + 0.3\mu_3 = 0.2$ e $\mathbf{C}\Sigma\mathbf{C}^{\mathbf{T}} = 0.31$ (cálculos logo abaixo em Python).

[25]:
```
b = np.array([[0.0, 0.0, 0.0]])
C = np.array([[0.4, 0.3, 0.3]])

C @ sigma @ C.T
```

[25]:
```
array([[0.31]])
```

- Distribuição de um indicador composto pelas 3 variáveis normalizadas: $T = 0.4(X_1 10)/2 + 0.3(X_2 20)/\sqrt{30} + 0.3(X_3 + 50)/\sqrt{94}$

Resposta: Tentaremos decompor o indicador T na forma $\mathbf{b} + \mathbf{C}\mathbf{X}$:

$$
\begin{aligned}
T &= \begin{pmatrix} 0.2 & \frac{0.3}{\sqrt{30}} & \frac{0.3}{\sqrt{94}} \end{pmatrix} \begin{pmatrix} X_1 - 10 \\ X_2 - 20 \\ X_3 + 50 \end{pmatrix} \\
&= \begin{pmatrix} 0.2 & \frac{0.3}{\sqrt{30}} & \frac{0.3}{\sqrt{94}} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix} + \begin{pmatrix} 0.2 & \frac{0.3}{\sqrt{30}} & \frac{0.3}{\sqrt{94}} \end{pmatrix} \begin{pmatrix} -10 \\ -20 \\ 50 \end{pmatrix} \\
&= \begin{pmatrix} 0.2 & \frac{0.3}{\sqrt{30}} & \frac{0.3}{\sqrt{94}} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix} + (-1.54831)
\end{aligned}
$$

Sendo assim, saberemos que $T \sim N_{k=1}(\mathbf{b} + \mathbf{C}\mu, \mathbf{C}\Sigma\mathbf{C}^{\mathbf{T}})$, onde $\mathbf{b} + \mathbf{C}\mu \approx -1.68643$ e $\mathbf{C}\Sigma\mathbf{C}^{\mathbf{T}} \approx 0.01309$ (cálculos logo abaixo em Python).

[26]:
```
# Calculando valor de b
b = np.array([[0.2, 0.3/np.sqrt(30), 0.3/np.sqrt(94)]]) @ np.array([[-10, -20,␣
↪50]]).T
b
```

[26]:
```
array([[-1.54831325]])
```

[27]:
```
# Calculando valor de b + C @ mu
C = np.array([[0.2, 0.3/np.sqrt(30), 0.3/np.sqrt(94)]])
b + C @ mu
```

[27]:
```
array([[-1.68642797]])
```

[28]:
```
# Calculando valor de C @ sigma @ C.T
C @ sigma @ C.T
```

[28]:
```
array([[0.01309709]])
```

- Distribuição conjunta de $(X_1 - X_2, 4X_1 + 2X_2 - X_3)$.

Resposta: Esse vetor segue uma distribuição normal multivariada com $k = 2$, podendo ser decomposto na forma **CX** onde

$$C = \begin{pmatrix} 1 & -1 & 0 \\ 4 & 2 & -1 \end{pmatrix}$$

Sendo assim, $T \sim N_{k=2}(\mathbf{C}\mu, \mathbf{C}\Sigma\mathbf{C}^{\mathbf{T}})$, onde $\mathbf{C}\mu$ e $\mathbf{C}\Sigma\mathbf{C}^{\mathbf{T}}$ são calculados logo abaixo em Python.

[29]:
```
C = np.array([
    [1, -1, 0],
    [4, 2, -1]
])

C @ mu
```

[29]:
```
array([[-1],
       [-6]])
```

[30]:
```
C @ sigma @ C.T
```

[30]:
```
array([[10, -2],
       [-2,  6]])
```

- Distribuição conjunta de $(X_1, aX_1 + bX_2 + cX_3)$, onde a, b, c são constantes reais. Em particular, encontre a covariância entre X_1 e o indicador $Y = aX_1 + bX_2 + cX_3$ formado pela combinação linear de X_1, X_2 e X_3.

Resposta: Esse vetor segue uma distribuição normal multivariada com $k = 2$, podendo ser decomposto na forma **CX** onde

$$C = \begin{pmatrix} 1 & 0 & 0 \\ a & b & c \end{pmatrix}$$

O vetor de médias será o seguinte:

$$\mathbf{C}\mu = \begin{pmatrix} 1 & 0 & 0 \\ a & b & c \end{pmatrix} \begin{pmatrix} -1 \\ 0 \\ 2 \end{pmatrix} = \begin{pmatrix} -1 \\ -a + 2c \end{pmatrix}$$

A matriz de covariância será:

$$
\begin{aligned}
\mathbf{C\Sigma C^T} &= \begin{pmatrix} 1 & 0 & 0 \\ a & b & c \end{pmatrix} \begin{pmatrix} 1 & -2 & 0 \\ -2 & 5 & 0 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} 1 & a \\ 0 & b \\ 0 & c \end{pmatrix} \\
&= \begin{pmatrix} 1 & -2 & 0 \\ a-2b & -2a+5b & 2c \end{pmatrix} \begin{pmatrix} 1 & a \\ 0 & b \\ 0 & c \end{pmatrix} \\
&= \begin{pmatrix} 1 & a-2b \\ a-2b & a^2-4ab+5b^2+2c^2 \end{pmatrix}
\end{aligned}
$$