DISTRIBUIÇÃO POISSON

- Teórica
  - Definição da v.a.
  - Notação
  - Parâmetros:
  - Função de probabilidade
  - Média
  - Variância
  - Função de distribuição acumulada
- Código Python
  - Biblioteca
  - Calcular X = x
  - Calcular X <= x
  - Calcular X > x
  - Calcular z < X <= x
- Extensão da classe "scipy"
- Exercícios
  - Exercício 24
    - 24.1) Poder pescar-se pelo menos um peixe.
    - 24.2) Poder pescar-se mais de um peixe, quando lá existe pelo menos um.
  - Exercício 25
    - 25.1) Calcule a probabilidade de, num dado dia, se enviar veículos para outro parque.
    - 25.2) Para permitir recolher todos os veículos que chegarem em pelo menos 98% dos dias, as instalações devem ser aumentadas para comportar, no mínimo, quantos veículos?
  - Exercício 28
    - 28.1) Calcule o número médio de avarias que ocorrem, por hora, no dispositivo referido.
    - 28.2) Qual a probabilidade de que nenhum deles avarie num período de 15 minutos?

Teórica

Definição da v.a.

X v.a. que representa o número de eventos que ocorrem nesse intervalo de tempo (ou nessa região).

Notação

$X \sim Po(\lambda)$
λ -> Número médio de eventos

Parâmetros:

$\lambda > 0$

Função de probabilidade

$$f(x) = \begin{cases} \frac{e^{-\lambda}\lambda^X}{X!}, x \in \{0, 1, 2, \dots\} \\ 0, \quad\quad , x \notin \{0, 1, 2, \dots\} \end{cases}$$

Média

$E(X) = \mu_X = \lambda$

Variância

$VAR(X) = \sigma_X^2 = \lambda$

Função de distribuição acumulada

$$f(x) = \begin{cases} 0, & X < 0 \\ e^{-\lambda} \sum_{i=0}^{Int[X]} \frac{\lambda^i}{i!}, & X \geq 0 \end{cases}$$

Código Python

Biblioteca

```
from scipy import stats
```

Calcular X = x

```
stats.poisson.pmf(x, mean)
```

Calcular X <= x

```
stats.poisson.cdf(x, mean)
```

Calcular X > x

```
1 - stats.poisson.cdf(x, mean)
```

Nota

Caso seja maior ou igual temos de calcular a probabilidade do número antes de x

Calcular z < X <= x

```
stats.poisson.cdf(x, mean) - stats.poisson.cdf(z, mean)
```

Nota

Caso seja maior ou igual temos de calcular a probabilidade de do número antes de z

Extensão da classe "scipy"

```
from scipy import stats

class PoissonExtension:
    def __init__(self, mean, spaceTimeQuantity):
        self.mean = mean
        self.spaceTimeQuantity = spaceTimeQuantity

    def Equal(self, x, toPercentage=False):
        return stats.poisson.pmf(x, self.mean)*(100 if toPercentage else 1)
    def LessOrEqual(self, x, toPercentage=False):
        return stats.poisson.cdf(x, self.mean)*(100 if toPercentage else 1)
    def Less(self, x, passe = 1, toPercentage=False):
        return stats.poisson.cdf(x-passe, self.mean)*(100 if toPercentage else
1)
    def Greater(self, x, toPercentage=False):
        return 1 - stats.poisson.cdf(x, self.mean)*(100 if toPercentage else
1)
    def GreaterOrEqual(self, x, passe = 1, toPercentage=False):
        return 1 - stats.poisson.cdf(x-passe, self.mean)*(100 if toPercentage
else 1)
    def GreaterAndLessOrEqual(self, a, b, toPercentage=False):
        return stats.poisson.cdf(b, self.mean) - stats.poisson.cdf(a,
self.mean)*(100 if toPercentage else 1)
    def GreaterOrEqualAndLessOrEqual(self, a, b, passe = 1,
```

```
toPercentage=False):
        return stats.poisson.cdf(b, self.mean) - stats.poisson.cdf(a-passe,
self.mean)*(100 if toPercentage else 1)
    def GreaterOrEqualAndLess(self, a, b, passe = 1, toPercentage=False):
        return stats.poisson.cdf(b-passe, self.mean) - stats.poisson.cdf(a,
self.mean)*(100 if toPercentage else 1)
    def GreaterAndLess(self, a, b, passe = 1, toPercentage=False):
        return stats.poisson.cdf(b-passe, self.mean) - stats.poisson.cdf(a-
passe, self.mean)*(100 if toPercentage else 1)

    def ScaleMean(self, targetSpaceTimeQuantity):
        return self.mean*targetSpaceTimeQuantity/self.spaceTimeQuantity

    @staticmethod
    def GetMeanOfProbabilityTable(self, probabilityTable):
        if(probabilityTable.length() != 2) : return None
        mean = 0
        for i in probabilityTable[0].length() :
            mean += probabilityTable[0][i]*probabilityTable[1][i]
```

- ir buscar a media pelo x e pela probabilidade
- obter o n pela probabilidade

Exercícios

Exercício 24

24.1) Poder pescar-se pelo menos um peixe.

$100dm^3 = 0.1m^3$
X: Número de peixes pescados em $0.1m^3$.
$X \sim Po(\lambda)$
$\lambda = 1$
$P(X \geq 1) = 1 - P(X = 0) = 1 - 0.368 = 0.632$

```
from scipy import stats
n = 0
media = 1
p0 = stats.poisson.pmf(n, media)
print(f"A probabilidade de x = 0 : {p0:.3f}")
print(f"A probabilidade de x > 0 : {(1 - p0):.3f}")
```

Source Code

A probabilidade de x = 0 : 0.368

A probabilidade de x > 0 : 0.632

24.2) Poder pescar-se mais de um peixe, quando lá existe pelo menos um.

$$P(X > 1|X \geq 1) = \frac{P(X>1 \cap X \geq 1)}{P(X \geq 1)} = \frac{P(X>1)}{P(X \geq 1)} = \frac{1-P(X \leq 1)}{P(x \geq 1)} = \frac{0.264}{0.632} = 0.418$$

```
n2 = 1
p_le1 = stats.poisson.cdf(n2, media)
print(f"A probabilidade de x <= 1 : {p_le1:.3f}")
print(f"A probabilidade de x > 1 : {(1-p_le1):.3f}")
print((1-p_le1)/(1-p0))
```

Source Code

A probabilidade de x <= 1 : 0.736

A probabilidade de x > 1 : 0.264

0.41802329313067355

Exercício 25

25.1) Calcule a probabilidade de, num dado dia, se enviar veículos para outro parque.

X: Número de veículos que chega a um parque em 1 dia.

$X \sim Po(\lambda)$

$\lambda = 2$

$P(X > 3) = 1 - P(X \leq 3) = 1 - 0.8571 = 0.1429$

```
from scipy import stats
n = 3
media = 2
p_le3 = stats.poisson.cdf(n, media)
print(f"A probabilidade de x <= 3 = {p_le3:.4f}")
print(f"A probabilidade de x > 3 = {(1 - p_le3):.4f}")
```

Source Code

A probabilidade de x <= 3 = 0.8571

A probabilidade de x > 3 = 0.1429

25.2) Para permitir recolher todos os veículos que chegarem em pelo menos 98% dos dias, as instalações devem ser aumentadas para comportar, no mínimo, quantos veículos?

Y: Número de veículos recolhidos no parque.
$Y \sim Po(\lambda)$
$\lambda = 2$
$P(Y \leq n) \geq 0.98 \Leftrightarrow F(n) \geq 0.98 \Leftrightarrow n \geq F^{-1}(0,98) \Leftrightarrow F^{-1}(0,98) = 5$

```
media = 2
p = 0.98
n_minimo = stats.poisson.ppf(p, media)
print(f"O numero minimo e de {n_minimo:.0f}")
```

Source Code
O numero mínimo e de 5

Exercício 28

$$f(x) = \begin{cases} 0, x \notin IN_0 \\ \frac{\lambda^x}{x!}e^{-\lambda}, x \in IN_0 \end{cases}$$

28.1) Calcule o número médio de avarias que ocorrem, por hora, no dispositivo referido.

X: Número de avarias por hora.
$X \sim Po(\lambda)$
$P(X = 1) = P(X = 2)$
$\frac{\lambda^1}{1!}e^{-\lambda} = \frac{\lambda^2}{2!}e^{-\lambda}$
$\Leftrightarrow \lambda = 2$

28.2) Qual a probabilidade de que nenhum deles avarie num período de 15 minutos?

Y: Número dispositivos que avariam em 5, ao longo de 15 minutos.
Z: Número de avarias em 15 minutos
$X \sim Bi(n, p)$
$n = 5$
$p = P(Z \geq 1) = 0.393$

```
p = 0.393
n = 5
```

```
x = 0
print(f"A probabilidade de x = 0 : {(stats.binom.pmf(x, n, p)):.3f}")
```

Source Code

A probabilidade de x = 0 : 0.082

$Z \sim Po(\lambda)$

$\lambda = 2 \times \frac{1}{4} = 0.5$

```
from scipy import stats
n = 0
media = 0.5
p0 = stats.poisson.pmf(n, media)
print(f"A probabilidade de x = 0 : {p0:.3f}")
print(f"A probabilidade de x > 0 : {(1 - p0):.3f}")
```

Source Code

A probabilidade de x = 0 : 0.607

A probabilidade de x > 0 : 0.393