Resumo das funções

Utilizando o Microsoft Excel, R e Python

Modelos probabilísticos

Distribuição Binomial

Utilizando o Microsoft Excel:

Distribuição Binomial para o cálculo de **P(X=x)**
=DISTR.BINOM(x;n;p;**FALSO**)

Distribuição Binomial para o cálculo de **P(0≤X≤x)**
=DISTR.BINOM(x;n;p;**VERDADEIRO**)*

*Os resultados exibidos mostrarão a probabilidade acumulada!

Distribuição Binomial

Utilizando o R:

Distribuição Binomial para o cálculo de **P(X=x)**
dbinom(x,size,prob,log = FALSE)

Distribuição Binomial para o cálculo de **P(0≤X≤x)**
pbinom(x,size,prob,**lower.tail=TRUE**,log = FALSE)

onde:
size = n
prob= p
lower.tail=TRUE → Fornece a probabilidade acumulada → P(X≤x)
lower.tail=FALSE → Fornece a probabilidade acima de x → P(X>x)
log=FALSE → Fornece as probabilidades numéricas e não em escala logarítmica.

Distribuição Binomial

Utilizando o Python:

Distribuição Binomial para o cálculo de **P(X=x)**
binom.pmf(x,n,p)

Distribuição Binomial para o cálculo de **P(0≤X≤x) = P(X≤x)**
binom.cdf(x,n,p)

Distribuição Binomial para o cálculo de **P(X>x)**
binom.sf(x,n,p)

Importante:
Para usar as funções de cálculo de probabilidade para a distribuição binomial no Python é necessário primeiramente que você importe a função *binom:*
from scipy.stats **import** binom

Distribuição Poisson

Utilizando o Microsoft Excel:

Distribuição Poisson para o cálculo de **P(X=x)**
=DIST.POISSON(x;lambda;**FALSO**)

Distribuição Poisson para o cálculo de **P(0≤X≤x)**
=DIST.POISSON(x;lambda;**VERDADEIRO**)*

*Os resultados exibidos mostrarão a probabilidade acumulada!

Distribuição Poisson

Utilizando o R:

Distribuição Poisson para o cálculo de **P(X=x)**
dpois(x, lambda, log = FALSE)

Distribuição Poisson para o cálculo de **P(0≤X≤x)**
ppois(x, lambda, **lower.tail = TRUE**, log.p = FALSE)

onde:
lower.tail=TRUE → Fornece a probabilidade acumulada → P(X≤x)
lower.tail=FALSE → Fornece a probabilidade acima de x → P(X>x)
log=FALSE → Fornece as probabilidades numéricas e não em escala logarítmica.

Distribuição Poisson

Utilizando o Python:

Distribuição Poisson para o cálculo de $P(X=x)$
poisson.pmf(x,media)

Distribuição Poisson para o cálculo de $P(0 \leq X \leq x) = P(X \leq x)$
poisson.cdf(x,media)

Distribuição Poisson para o cálculo de $P(X>x)$
poisson.sf(x,media)

Importante:
Para usar as funções de cálculo de probabilidade para a distribuição binomial no Python é necessário primeiramente que você importe a função *poisson:*
from scipy.stats **import** poisson

Distribuição Exponencial

Utilizando o Microsoft Excel:

Distribuição exponencial para o cálculo de P(0<X<x)
=DISTR.EXPON(x;alfa;VERDADEIRO)*

*Os resultados exibidos mostrarão a probabilidade acumulada!

Distribuição Exponencial

Utilizando o R:

Distribuição exponencial para o cálculo de $P(0<X\leq x)$
pexp(x, rate = alfa, lower.tail = TRUE, log.p = FALSE)

onde:
alfa= parâmetro da distribuição exponencial
lower.tail=TRUE → Fornece a probabilidade acumulada → $P(X\leq x)$
lower.tail=FALSE → Fornece a probabilidade acima de x → $P(X>x)$
log=FALSE → Fornece as probabilidades numéricas e não em escala logarítmica.

Distribuição Exponencial

Utilizando o Python:

Distribuição Exponencial para o cálculo de **$P(0 \leq X \leq x) = P(X \leq x)$**
expon.cdf(x, *scale=media*)

Distribuição Exponencial para o cálculo de **$P(X>x)$**
expon.sf(x, *scale=media*)

Onde:
alfa= parâmetro da distribuição exponencial que representa o inverso da média.
media = 1/alfa

Importante:
Para usar as funções de cálculo de probabilidade para a distribuição exponencial no Python é necessário primeiramente que você importe a função expon*:*
from scipy.stats **import** expon

Distribuição Normal

Utilizando o Microsoft Excel:

Distribuição normal para o cálculo de P(X<x)
=DIST.NORM.N(x;media;desvio padrão;VERDADEIRO)*

*Os resultados exibidos mostrarão a probabilidade acumulada!

Cálculo inverso: Informa o valor de x a partir de uma probabilidade acumulada
=INV.NORM.N(probabilidade;media;desvio padrão)

Distribuição Normal

Utilizando o R:

Distribuição normal para o cálculo de $P(X \leq x)$
pnorm(x, mean = m, sd = s, lower.tail = TRUE, log.p = FALSE)

onde:
m= média
s= desvio padrão
lower.tail=TRUE → Fornece a probabilidade acumulada → $P(X \leq x)$
lower.tail=FALSE → Fornece a probabilidade acima de x → $P(X>x)$
log=FALSE → Fornece as probabilidades numéricas e não em escala logarítmica.

Cálculo inverso: Informa o valor de x a partir de uma **probabilidade acumulada**
qnorm(p, m, s, lower.tail = TRUE)

onde:
p → representa a probabilidade acumulada até x
m→ média
s → Desvio padrão

Distribuição Normal

Utilizando o Python:

Distribuição Normal para o cálculo de probabilidade $P(X \leq x)$
norm.cdf(x, m, s)

Distribuição Normal para o cálculo de probabilidade $P(X > x)$
norm.sf(x, m, s)

onde:
m= média
s= desvio padrão

Cálculo inverso: Informa o valor de x a partir de uma **probabilidade acumulada**
norm.ppf(p, m, s)

onde: p → representa a probabilidade acumulada até x
m→ média
s → Desvio padrão

Importante:
Para usar as funções de cálculo de probabilidade para a distribuição normal no Python é necessário primeiramente que você importe a função norm*:*
from scipy.stats **import** norm