

chapter_02

April 1, 2022

1 0.0. Imports

1.1 0.1. Julia & Python Imports

```
[75]: using CSV;
      using PyCall;
      using Printf;
      using StatsBase;
      using DataFrames, FreqTables
      using HypothesisTests

      include("../scripts/utils_2.jl")
      np = pyimport("numpy");
      sns = pyimport("seaborn");
      ss = pyimport("scipy.stats");
      wrg = pyimport("warnings");
      pd = pyimport("pandas");
      gs = pyimport("matplotlib.gridspec");

      prepro = pyimport("sklearn.preprocessing");
      md_sel = pyimport("sklearn.model_selection");
      models = pyimport("tensorflow.keras.models");
      layers = pyimport("tensorflow.keras.layers");
```

```
[3]: py"""
      import numpy as np
      def test(diff, diff_mean):
          return np.mean(diff > diff_mean)
      """

      wrg.filterwarnings("ignore")
```

2 2.0. Capítulo 2

2.1 2.1. Distribuição de Amostragem de uma Estatística

” A distribuição de uma estatística amostral como a média costuma ser mais regular e campanular do que a distribuição dos próprios dados quanto maior a amostra em que a estatística se baseia.

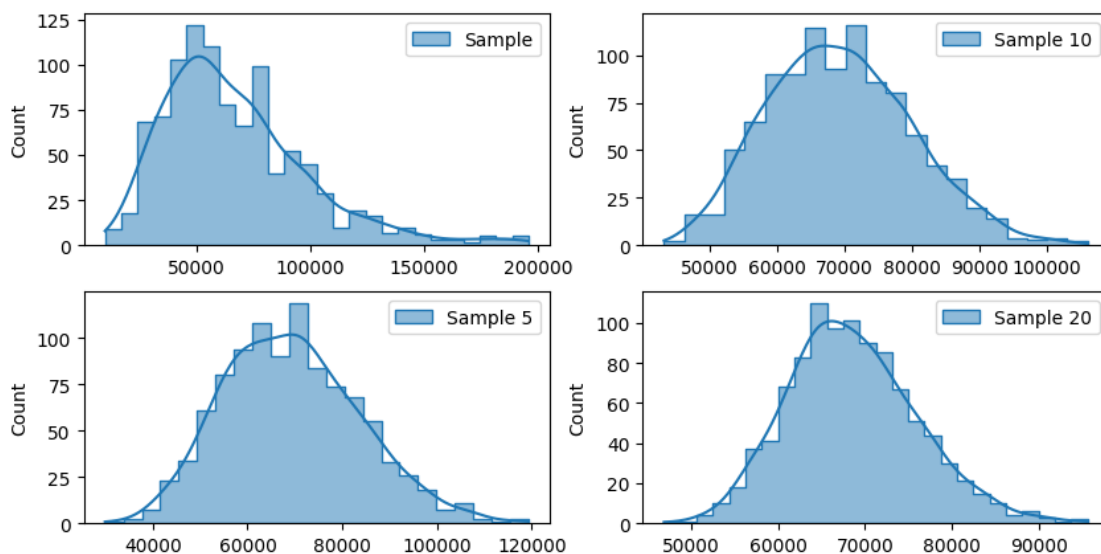
Além disso, quanto maior a amostra, mais estreita é a distribuição da estatística amostral. “Tende a Normal” ”

```
[5]: df = pd.read_csv("../data/loans_income.csv"); # Testando o Pandas DataFrame em Julia
```

```
[46]: sample = df.sample(1000)
sample_of_5 = pd.DataFrame( Dict("income" => [df["x"].sample(5).mean() for _ in 1:1000], "type" => "mean_of_5" ) )
sample_of_10 = pd.DataFrame( Dict("income" => [df["x"].sample(10).mean() for _ in 1:1000], "type" => "mean_of_10" ) )
sample_of_20 = pd.DataFrame( Dict("income" => [df["x"].sample(20).mean() for _ in 1:1000], "type" => "mean_of_20" ) )

result = pd.concat([sample, sample_of_5, sample_of_10, sample_of_20]);

fig, ax = plt.subplots(2, 2, figsize=(10, 5))
for k in zip([sample, sample_of_5, sample_of_10, sample_of_20], ["Sample", "Sample 5", "Sample 10", "Sample 20"], [1, 2, 3, 4]):
    sns.histplot( k[1], kde=True, element="step", ax=ax[k[3]], label=k[2] );
    ax[k[3]].legend()
end;
```



2.2. O Bootstrap

O bootstrap é uma forma eficiente e eficaz de estimar a distribuição amostral de uma estatística ou de parâmetros de modelo. Conceitualmente pode-se imaginar o Bootstrap como uma replicação da amostra original várias vezes de modo a ter uma população hipotética que representa todo o conhecimento da amostra original só que maior. Logo amostramos com reposição, dessa forma

cria-se efetivamente uma população infinita na qual a probabilidade de um elemento ser extraído continua a mesma de extração por extração. Com os resultados é possível encontrar um intervalo de confiança.

2.2.1 2.2.1. Sem Bootstrap

```
[61]: # Load and Prepare Dataset
df = DataFrame(CSV.File("../data/diabetes.csv"));
```

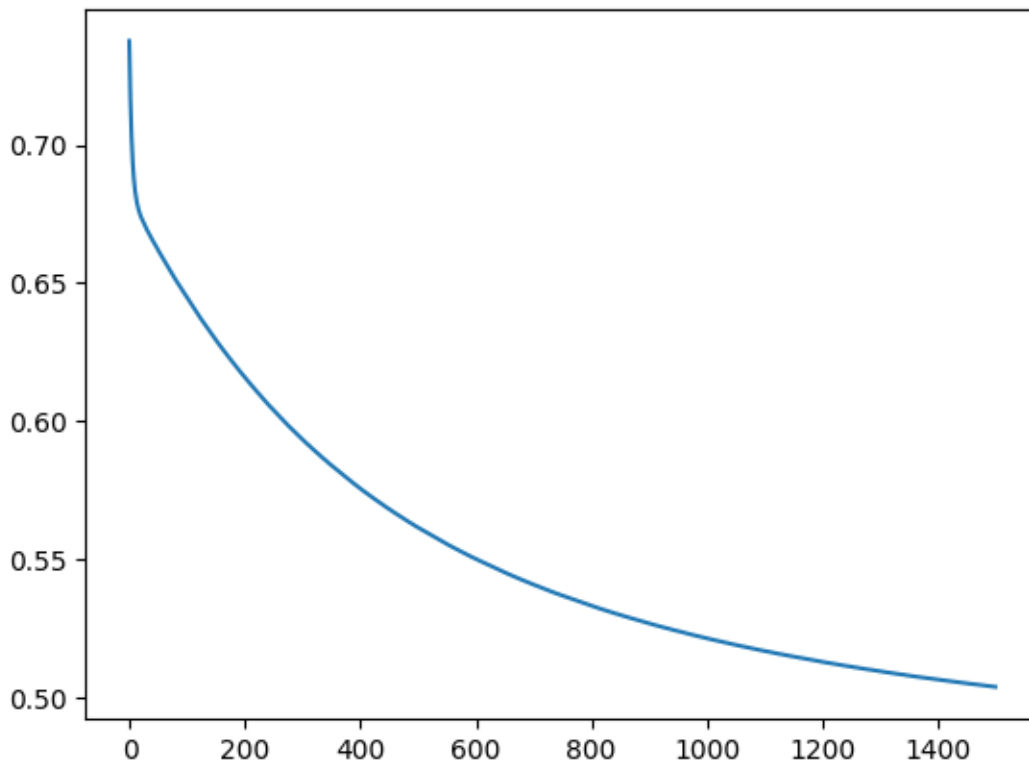
```
[59]: x_train, x_test, y_train, y_test = prepare_dataset(df);
```

```
[67]: # Simple Logistic Regression
model = models.Sequential()
model.add( layers.Dense( 1, activation="sigmoid" ) )
model.compile( optimizer="sgd", loss="binary_crossentropy",
↪ metrics=["accuracy"] )
```

```
[68]: history = model.fit( x_train, y_train, epochs=2000, verbose=0 );
test = model.evaluate( x_test, y_test );
```

```
3/3 [=====] - 0s 3ms/step - loss: 0.5622 - accuracy:
0.7662
```

```
[117]: plt.plot(history.history["loss"]);
```



Conclusão

```
[69]: @printf "Acurácia do modelo é %.2f%%" maximum(history.history["accuracy"])*100
```

Acurácia do modelo é 77.28%

Conclusão bem simples em relação ao modelo especificamente e não em relação ao negócio para exemplificar a próxima ferramenta.

2.2.2 2.2.1. Com Bootstrap

Etapas do Bootstrap

1. Definir a “cabeça” do teste que são:
 1. N de iterações.
 2. Comprimento da Amostragem. (Por Exemplo 50%)
2. Iniciando o Experimento.
 1. Definir a amostra aleatória.
 2. Separar em treino e teste e fazer a preparação dos dados. OBS: Fazer a preparação depois para evitar vazamento de dados” ”
 3. Treinar algum Modelo, no caso escolhi uma DecisionTree.
 4. Avaliar o modelo e guardar a métrica do modelo que esta a estimar, ex: Acurácia.

```
[25]: # Configuration of Bootstrap
n_inter = 1000
n_size = trunc(Int, (length(df.Age) * 0.5))
stats = []

# Set Up Data
df = hcat(df[:, 1], df[:, 2], df[:, 3], df[:, 4], df[:, 5], df[:, 6], df[:, 7],
↳df[:, 8], df[:, 9])

for i in 1:n_inter
    sample = utils.resample( df, n_samples=n_size );
    y = sample[:, 9];
    x = sample[:, 1:8];

    x = mms.fit_transform( x );

    # Split Dataset
    x_train, x_test, y_train, y_test = model_selection.train_test_split( x, y,
↳train_size=0.9 );

    model = tree.DecisionTreeClassifier() # Decision Tree, dont NN.
    model.fit( x_train, y_train )

    prediction = model.predict( x_test )
```

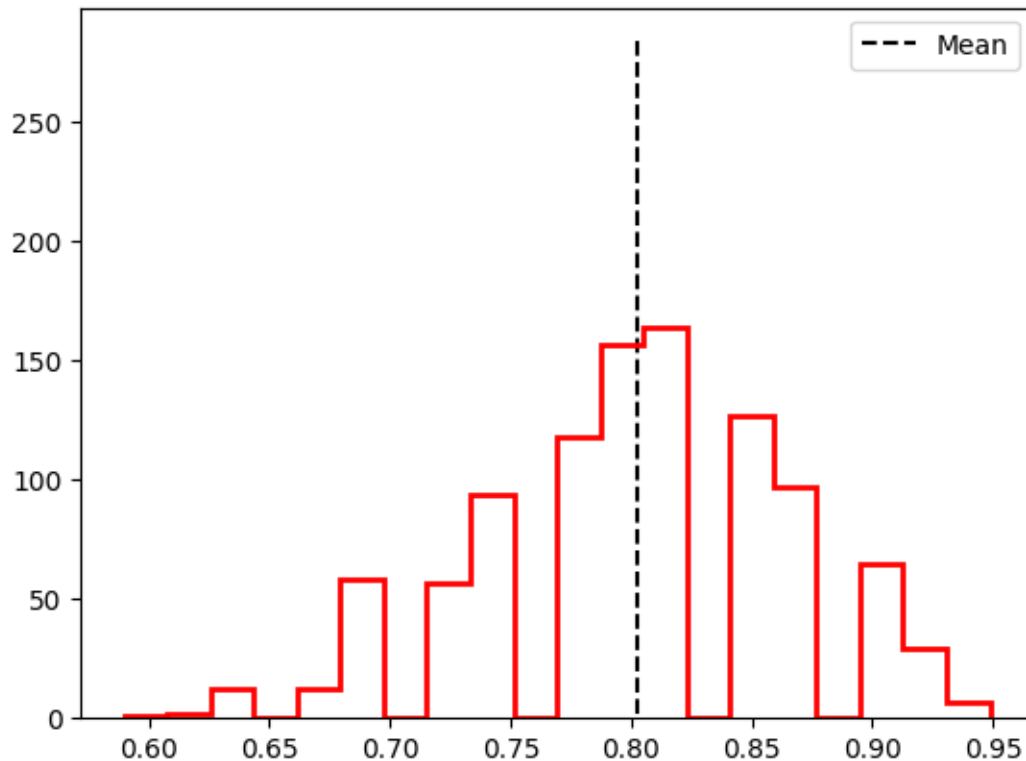
```

score = metrics.accuracy_score( y_test, prediction )

append!(stats, score)
end

```

```
[106]: plot_bootstrap( stats, "step", "r" )
```



Conclusão

```
[110]:
= 0.95

p = ((1.0 - )/2.0)*100
lower = np.percentile(stats, p)*100

p = ( + (1.0 - )/2.0) *100
max = np.percentile(stats, p)*100

@printf "%.0f%% Confidence Intervals %.2f%% and %.2f%%" *100 lower max

```

95% Confidence Intervals 66.67% and 92.31%

Você tem 95% de confiança que a acurácia do modelo esta entre 66.67 e 92.31 com as configurações de parametros do modelo, e tamanho da amostra, geralmente com amostras maiores, tende a ter

uma acurácia maior pois você tem mais exemplos de treinamento. Basta alterar o α para conseguir outros intervalos.

2.3. Distribuições

2.3.1 Distribuição Binomial

A distribuição Binomial é, vamos dizer assim uma continuação da distribuição de Bernoulli, onde a distribuição de Bernoulli trabalha somente com duas possibilidades.

1 geralmente chamado de evento de sucesso. 0 geralmente é chamado de fracasso.

```
[17]: bernoulli = pd.DataFrame([["Azul", 0.35], ["Vermelho", 0.65]]);
      bernoulli.columns = ["X", "Probabilidade"];
      bernoulli
```

```
[17]: PyObject          X  Probabilidade
0      Azul            0.35
1  Vermelho            0.65
```

É a distribuição Binomial é vamos dizer assim, uma continuação da distribuição de bernoulli, conforme eu repita o ensaio de bernoulli n vezes, com a mesma probabilidade p de sucesso caracteriza uma distribuição Binomial.

Critérios para a distribuição Binomial 1. O experimento é repetido n vezes, onde cada tentativa é independente da outra. 2. Somente dois resultados possíveis, a o resultado de interesse e o resultado de fracasso.

Formula da Distribuição Binomial

$$P(r/n, p) = \binom{n}{r} \cdot p^r \cdot (1-p)^{n-r}, \text{ onde } \binom{n}{r} = C_{nk} = \frac{n!}{r! \cdot (n-r)!}$$

- r : Eventos de Interesse.
- n : Repetições.
- p : Probabilidade do Evento.

A média da distribuição binomial é dada pela formula: $n \cdot p$ A variância da distribuição binomial é dada pela formula: $\sqrt{n \cdot p \cdot (1-p)}$

Na formula da distribuição Binomial também pode se trabalhar com um conceito chamado de “Complementar”, que nada mais é que uma forma mais rápida para calcular a binomial quando se tem um intervalo de eventos maior que um. $P(x \geq k) = 1 - P(x < k)$ $P(x > k) = 1 - P(x \leq k)$ $P(x \leq k) = 1 - P(x > k)$ $P(x < k) = 1 - P(x \geq k)$ Por exemplo se você for calcular um $n = 6$ para uma possibilidade de, digamos $P(x \geq 2)$, você teria que calcular 5 vezes com a fórmula, logo $P(x = 2) + P(x = 3) + P(x = 4) + P(x = 5) + P(x = 6)$ Porém, aplicando a regrazinha, inverte-se, logo: $1 - (P(x = 0) + P(x = 1))$

```
[16]: P(r, n, p) = (factorial(n) / (factorial(r) * factorial(n-r))) * (p^r) * ((1 -
      ↪p)^(n-r));
```

Qual é a probabilidade de sair 3 caras em 4 jogadas e a probabilidade de cada cara é $\frac{1}{2}$

$$P(2/4, \frac{1}{2}) = 6 \cdot (\frac{1}{2})^2 \cdot (\frac{1}{2})^2 = \frac{6}{16}$$

```
[18]: @printf "A probabilidade de sair três caras em quatro jogadas é: %.3f%% " P(2, 4, .5)*100
```

A probabilidade de sair três caras em quatro jogadas é: 37.500%

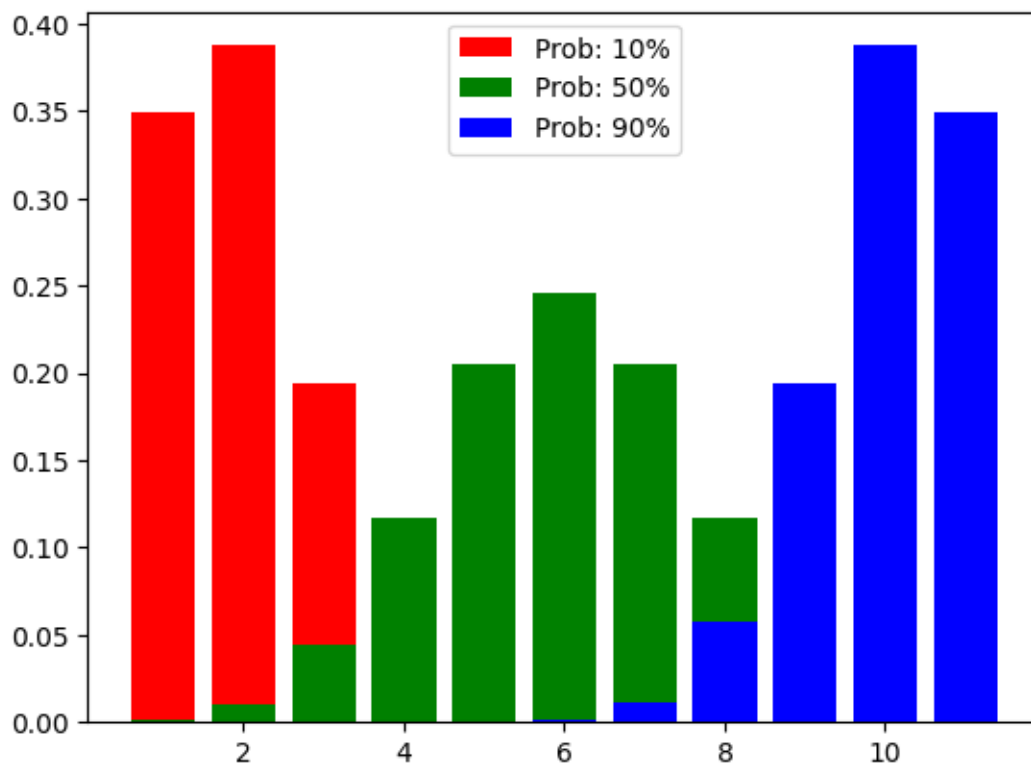
Qual é a chance de 3 pessoas que eu ligo das 10 entrarem em churn sabendo que a probabilidade de uma pessoa em churn na base de dados é 0.15%?

```
[19]: @printf "A probabilidade de 3 das 10 pessoas ligadas entrarem em churn é: %.3f%%" P(3, 10, .15)*100
```

A probabilidade de 3 das 10 pessoas ligadas entrarem em churn é: 12.983%

Conforme a probabilidade tende a o equilíbrio, ou seja, .5% de cair cara ou coroa, logo a distribuição parece uma Normal.

```
[194]: plot_three_bn( [0.1, 0.5, 0.9], ["r", "g", "b"], ["Prob: 10%", "Prob: 50%", "Prob: 90%"] )
plt.savefig("Binomial.png")
```



Mais qual é a relevância disso para ciência de dados?

Antes de mais nada, a distribuição binomial é uma boa pedida quando se tem poucos dados(Requer Testes de Potência de Amostras)” mas resumidamente, 200 amostras, pois, quando maior a amostra, mais esses testes são substituídos por modelo de ml.

Seguindo uma abordagem mais frequentista, segue o exemplo.

1. Se a probabilidade de conversão em relação a um telefonema é de 0.15, qual é a probabilidade de observar 3 conversões em 50 ligações ?

```
[126]: @printf "A probabilidade de exatamente 3 conversões em 50 ligações é de %.3f%%\n"
        ss.binom.pmf(3, 50, .15)
        @printf "A probabilidade de 3 ou menos conversões em 50 ligações é de %.3f%%\n"
        ss.binom.cdf(3, 50, .15)
```

A probabilidade de exatamente 3 conversões em 50 ligações é de 0.032%

A probabilidade de 3 ou menos conversões em 50 ligações é de 0.046%

Em outras palavras, é possível motivar ou *desmotivar* o time de marketing em relação a o telefonema dado os resultados de um teste binomial, por exemplo, se a conversão em 500 ligações é de > 40 em relação a 3 ou menos casos de sucesso, logo, qual é o custo dessas ligações, vale a pena realmente fazer 500 ligações ?

```
[101]: BinomialTest(2, 5, .01) # Sumário em Julia do teste Binomial
```

```
[101]: Binomial test
-----
Population details:
  parameter of interest:   Probability of success
  value under h_0:         0.01
  point estimate:          0.4
  95% confidence interval: (0.05274, 0.8534)

Test summary:
  outcome with 95% confidence: reject h_0
  two-sided p-value:         0.0020

Details:
  number of observations: 5
  number of successes:    2
```

A distribuição binomial também é utilizada em testes A/B para cliques em páginas web, mas, algoritmos de reinforcement learning já são possivelmente melhores para esse tipo de teste A/B em questão de botões, o algoritmo treina em tempo de execução e ao decorrer do treino, vai mostrar apenas o botão que tiver mais cliques.

2.3.2 2.3.2 Distribuição de Poisson

Alta concentração de eventos próximos ao eixo y, uma das principais características é que não tem repetições como na distribuição binomial trabalha em um intervalo contínuo.Ex: Em um estudo

de chuva ou cliques em um site, no exemplo da chuva, qual é a chance de uma chuva, só que não existe o evento “não chuva” entre duas chuvas.

Imagine um intervalo, que começa em 0 até uma variável W por exemplo. E eu divido em n intervalos muito pequenos, onde n tende ao infinito., logo a probabilidade está tendendo a 0 pois existem n intervalinhos, com essa quantidade de intervalos, virou uma binomial, ou seja, choveu ou não por exemplo.

$$P(r/\frac{\lambda}{n}, n) = \lim_{n \rightarrow \infty} (\frac{\lambda}{n})^r \cdot (1 - \frac{\lambda}{n})^{n-r} \cdot \frac{n!}{r!(n-r)!}$$

Distribuição de Poisson, logo λ (Quantidade de Chuva) = $p * n$, então $p = \frac{\lambda}{n}$. No Limite que n tende ao infinito, o produto de n e r não vai mudar pois r sempre vai ficar menor e n sempre vai ficando maior.

$$P(r/\lambda) = \frac{e^{-\lambda} \cdot \lambda^r}{r!}$$

A média e a variância da distribuição binomial é dada pelo: λE o Desvio Padrão é $\sqrt{\lambda}$

```
[132]: P( , r) = np.e^- * ^r / np.math.factorial(r);
```

Dado que eu esperava em média 35 carros entrando no shopping, qual a probabilidade de aparecer 20?

```
[143]: @printf "A probabilidade de 30 carros, sendo que eu esperava 20 é de %.3f%% \n"
        ↪ss.poisson.pmf(35, 20)
@printf "A probabilidade de 30 carros ou menos sendo que eu esperava 20 é de %.
        ↪3f%%" ss.poisson.cdf(35, 20)
```

A probabilidade de 30 carros, sendo que eu esperava 20 é de 0.001%

A probabilidade de 30 carros ou menos sendo que eu esperava 20 é de 0.999%

```
[141]: @printf "A probabilidade de 35 carros, sendo que esperava 20 é de: %.3f%%"
        ↪P(35, 20)
```

A probabilidade de 35 carros, sendo que esperava 20 é de: -0.000%

```
[140]: # Julia tem problema com elevar x a o espoeinte y
function f(x, y)
    [x*=x for _ in 1:y]
end
f(35, 20)
```

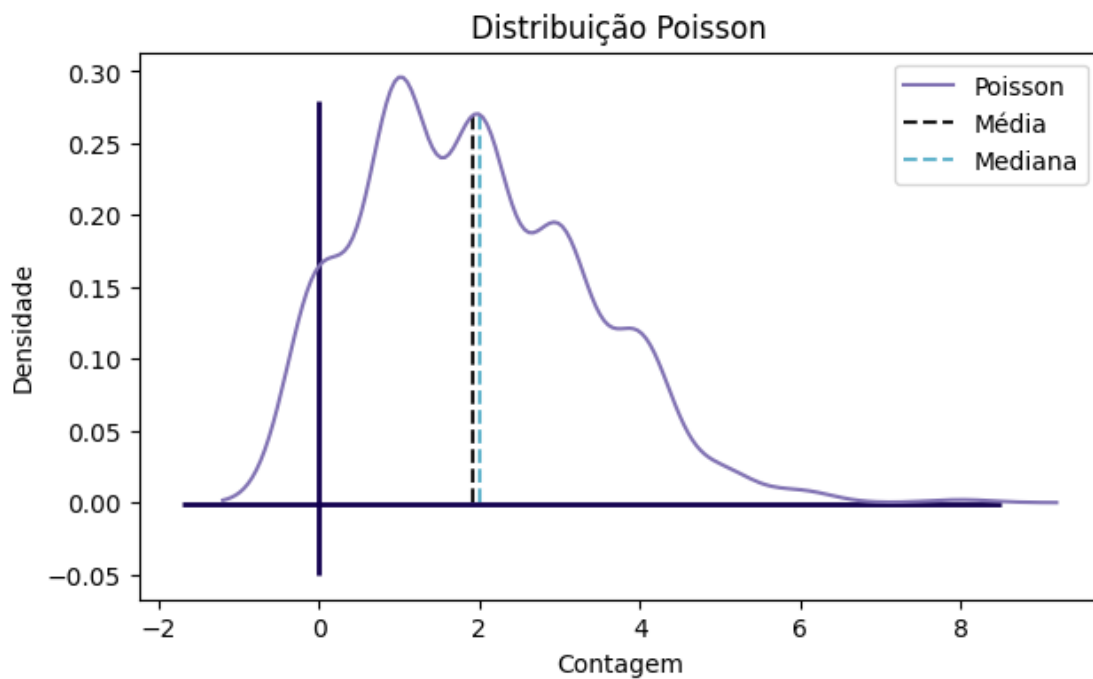
```
[140]: 20-element Vector{Int64}:
         1225
        1500625
       2251875390625
      6616016035436858689
```

```
7865930784382691969
-4822766768660441855
-1652024524321314303
 450275795304469505
-6392656039275616255
 7551947002216534017
 3654036140188672001
 8358544585278177281
 4785494631104806913
-2679742982427181055
 2893676706708193281
 8010019347241369601
  -48246705104617471
-5786811055723249663
-8394158839848501247
 541221425122377729
```

Logo em Python, aplicando a mesma função irá retornar a probabilidade de 0.0019

```
[895]: x = ss.poisson.rvs(2, size=500);
```

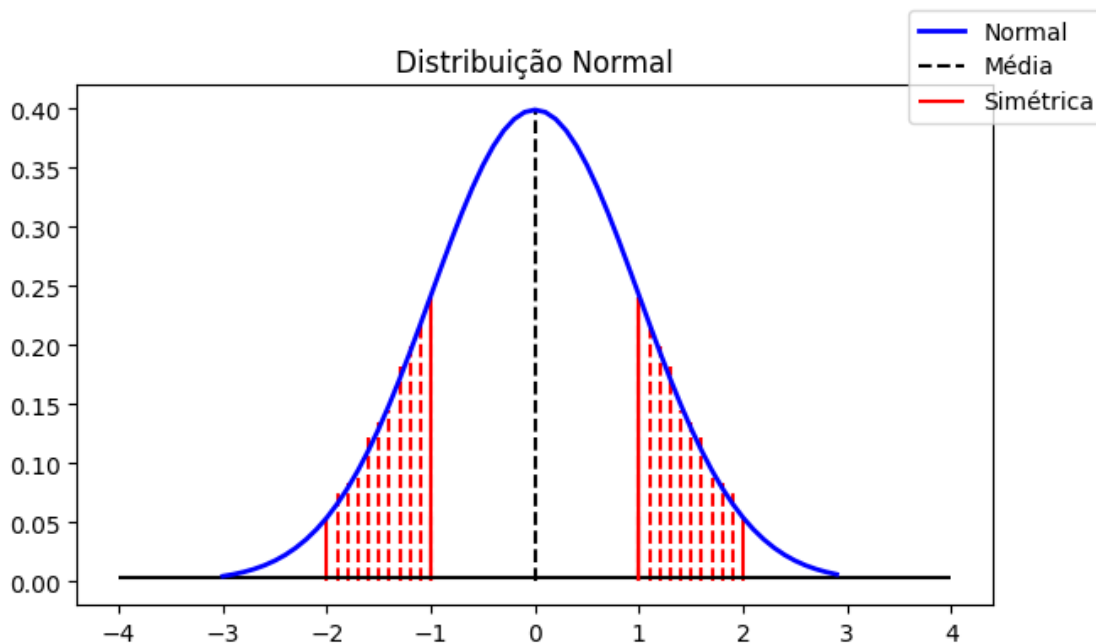
```
[896]: plot_poisson( x, "m" )
```



2.3.3 2.3.5 Distribuição Normal

A distribuição Normal é simétrica a média e as outras distribuições são geralmente moldadas de forma normal. Em uma distribuição normal 68% dos dados ficam dentro de um desvio-padrão da média e 90% dos dados em dois desvios-padrões. A diferença entre a distribuição normal das outras distribuições (binomial e poisson) é que na noção de distribuição discreta e contínua, ambas são distribuições discretas pois as possibilidades dos eventos eram discretos, agora x pode assumir uma probabilidade, logo a função é chamada de densidade de probabilidade. Onde para calcular a área em baixo da curva usa-se a ferramenta de Integral. $\int_0^1 f(x) dx$ Ou utiliza a tabela da normal. Na Integral, o primeiro valor de baixo (0) é o primeiro valor da esquerda para direita na distribuição, e o valor de cima (1) é justamente até aonde vai a área.

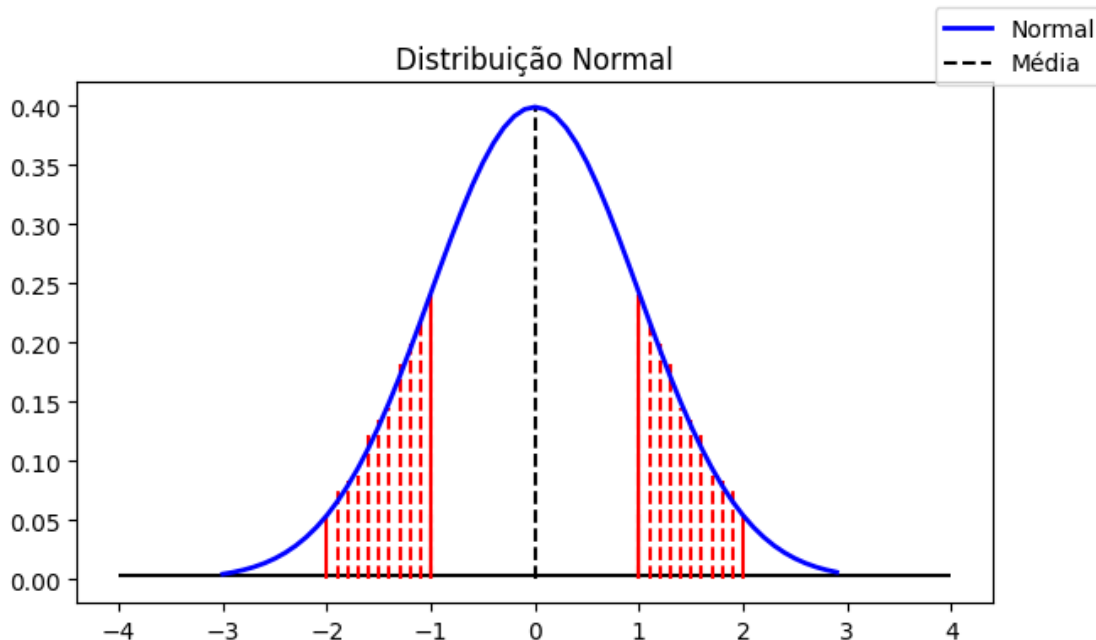
```
[97]: range = np.arange(-3, 3, 0.1)
      plot_normal( range, true )
```



Função densidade de probabilidade

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \cdot e^{\frac{-(x-\bar{x})^2}{2\sigma^2}}$$

```
[290]: plot_normal( range, true );
```



Como a variável x é contínua, sendo assim pode assumir infinitos valores, e também toda a área da curva gaussiana é 1^* . Logo para calcular a área entre 1 e 2 da normal:

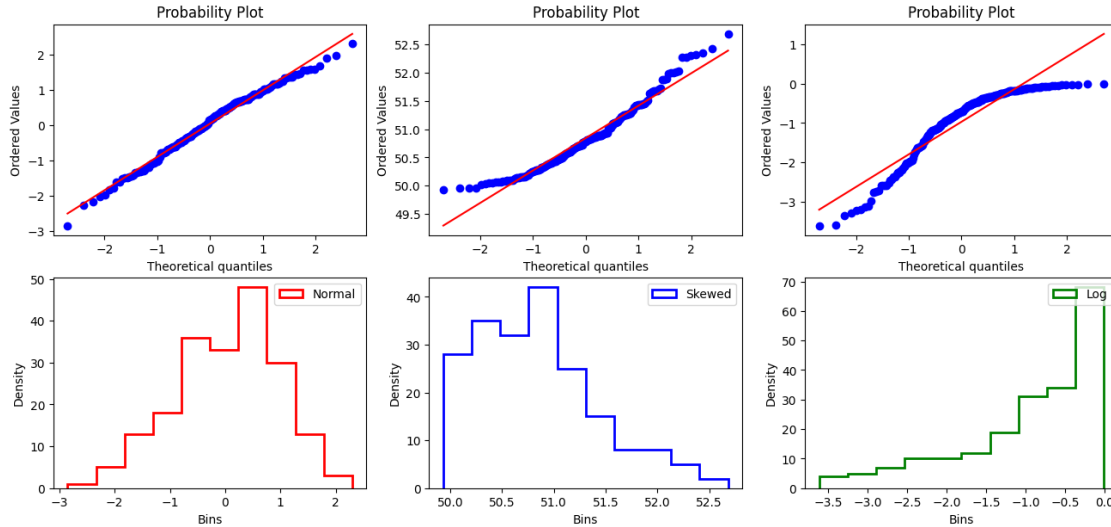
$$P = \int_1^2 \frac{1}{\sqrt{2\pi}\sigma} \cdot e^{\frac{-(x-\bar{x})^2}{2\sigma^2}} dx$$

2.3.4 2.3.6. QQ Plot

QQ plot nada mais é que um plot para visualizar como está o shape da distribuição, se tem skewness ou kurtosis. **Skewness:** Geralmente se fala em Skewness quando a distribuição está tombada para algum dos lados, uma Skewness **positiva** significa que a distribuição está mais deslocada para a esquerda, e uma skewness **negativa** quando está mais deslocada a direita, o exemplo na imagem da distribuição exponencial abaixo. **Kurtosis:** Mede o quanto a distribuição está esticada para cima ou com formação de longas caldas, quando mais pontuda, maior a kurtosis e quando mais normal, menor a kurtosis, o exemplo é a distribuição exponencial que tem uma certa kurtosis positiva.

```
[161]: # Existe um pacote chamado Plots que faz a mesma função de plotar o QQ Plot.
x = np.random.randn(200);
y = [log(abs(p)) for p in np.random.random(200)];
z = ss.skewnorm.rvs(a=10, loc=50, size=200)

plot_qq(x, y, z, "step")
plt.savefig("QQplot.png")
```



2.3.5 2.3.7. Normalização

A normalização é um conceito utilizado principalmente para treinar modelos de machine learning que consiste em movimentar a distribuição para o centro com média 0, resumidamente subtrair a média de todos os dados. Se reparar nos gráficos da distribuição normal, logo a média já está no 0. Se eu somar a média da distribuição em toda a distribuição ela vai ser deslocada para direita, se subtrair ela é deslocada a esquerda. E quando se divide por σ , logo a média é 0 e a dispersão é 1.

Quando esta normalizada é possível utilizar a tabela da normal padrão paara calcular a área em baixo da curva.Exemplo, dado uma média de 200 e desvio padrão de 4, qual é a $P(x > 210)$? - 1º Passo, calcular o Z, que nada mais é que subtrair a média e dividir pelo desvio padrão. Ou seja $z = \frac{210-200}{4} = 2.5$, esse é o resultado que deve ser encontrado a área, para isso so checara tabela, onde são 2,5 os dois primeiros números e 0 o terceiro número, o resultado vai ser 0.4938 - 2º Passo, Realizar a seguinte expressão $(.5 - .4938) \cdot 100$ Subtrair pela metade da distribuição normal o resultado para pegar somente a probabilidade de ser maior que 210, e multiplicar por 100 para deixar em porcentagem, logo o resultado final é: 0.62.

A Relevância para Ciência de Dados

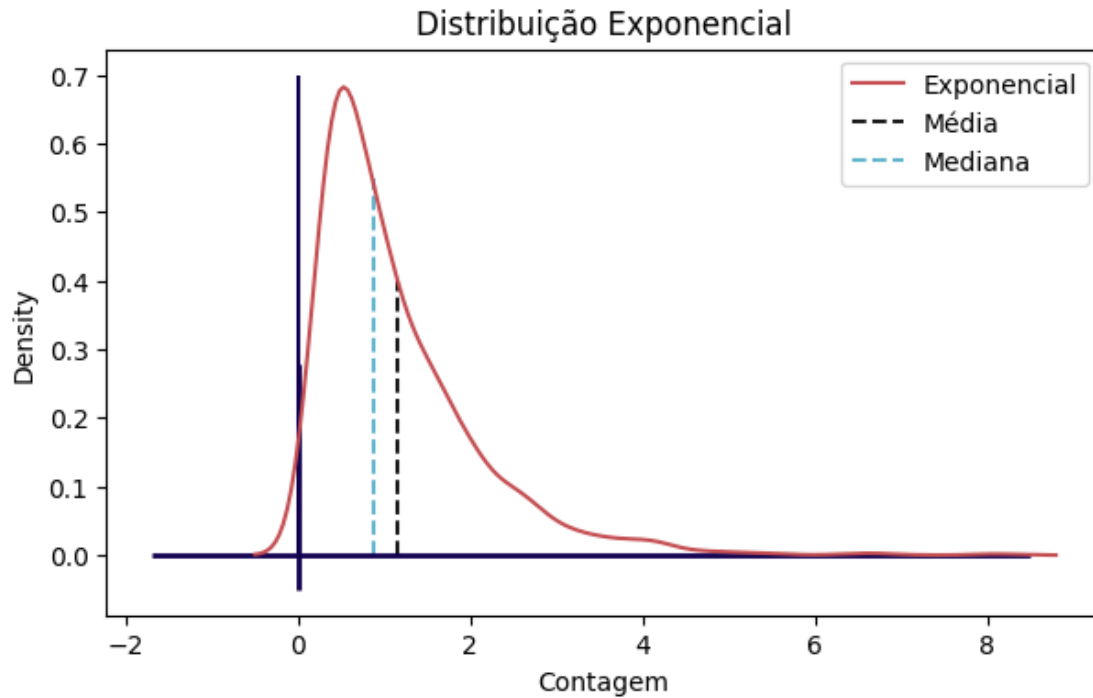
MUITAS, desde epserar uma distribuição normal dos erros de um modelo de regressão, até análise de dados para escolher qual tipo de técnica de normalização você vai utilizar.

2.3.6 2.3.8 Distribuição Exponencial

Usa o mesmo parâmetro λ da distribuição de Poisson, esse parâmetro permanece constante ao longo do período sendo considerado.É utilizado na engenharia para modelar falhas , tempo de visitas de sites, etc.

```
[870]: x = ss.expon.rvs( 0.2, size=1000);
```

```
[871]: plot_exp(x, "r")
```



2.3.7 2.3.9 Distribuição Weibull

É uma extensão da distribuição Exponencial, na qual a taxa de evento pode mudar de acordo com um “parâmetro de forma”. Se $\alpha > 1$, a probabilidade de um evento aumenta com o tempo. Se $\alpha < 1$, a probabilidade de um evento diminui com o tempo. Quando o α da distribuição de Weibull é 1, retorna a distribuição exponencial. Sendo assim, pode ser utilizada na análise de sobrevivência & confiabilidade, e sua função é:

Comulativa:

$$F(x, \alpha, \beta) = 1 - e^{-\left(\frac{x}{\beta}\right)^\alpha}$$

Densidade de Probabilidade:

$$f(x, \alpha, \beta) = \frac{\alpha}{\beta^\alpha} \cdot x^{\alpha-1} \cdot e^{-\left(\frac{x}{\beta}\right)^\alpha}$$

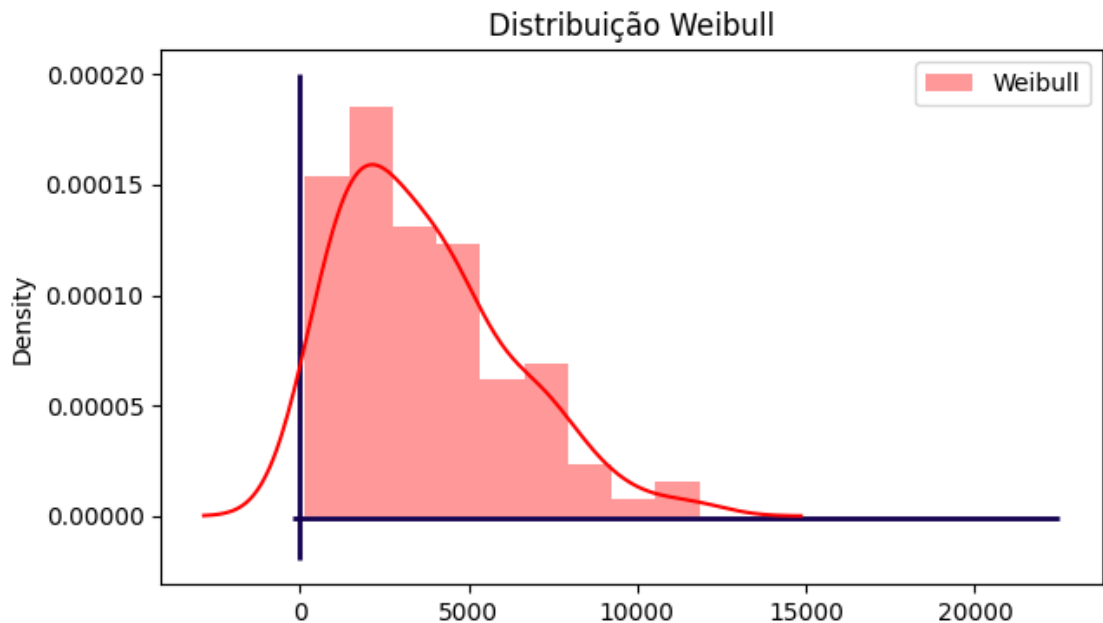
```
[276]: f(x, a, b) = (a / (b^a)) * (x^(a-1)) * ^(-(x/b)^a)
f(3, 3, 3)
```

```
[276]: 0.36787944117144233
```

```
[275]: 20^30 # não é possível utilizar números grandes pois o mesmo problema.
```

```
[275]: -8070450532247928832
```

```
x = ss.weibull_min.rvs(1.5, scale=5000, size=100)
plot_wei(x, "r")
```



3 x.0. Referências

PETER BRUCE & ANDREW BRUCE **Estatística prática para cientistas de dados: 50 conceitos essenciais.** Link: <https://www.amazon.com.br/Estat%C3%ADstica-Pr%C3%A1tica-Para-Cientistas-Dados/dp/855080603X> DAVID MATOS 8

Conceitos Estatísticos Fundamentais Para Data Science. Link: <https://www.cienciaedados.com/8-conceitos-estatisticos-fundamentais-para-data-science/> IGOR SOARES

Correlação não implica em Causalidade. Link: <https://medium.com/@felipemaiapolo/correla%C3%A7%C3%A3o-n%C3%A3o-implica-em-causalidade-8459179ad1bc> annahaensch

Número de Casos de Divórcio em Maine Link: <https://blogs.ams.org/blogonmathblogs/2017/04/10/divorce-and-margarine/> Wikipédia

Cramer's χ^2 Link: https://en.wikipedia.org/wiki/Cram%C3%A9r%27s_%5Bchi%5E2%5D VBURKEYACADEMY

What are Skewness and Kurtosis? Link: <https://www.youtube.com/watch?v=lK7nLzxiAQQ> (Discourse)

qqnorm & qqplot Link: <https://discourse.julialang.org/t/qqnorm-and-qqplot/6118/8> Professor Guru

Tabela Normal Padrão Link: <https://professorguru.com.br/tabela-normal.html>