Análise Estatística dos Retornos Diários do Índice Ibovespa:

Comparação entre Distribuições Normal e t de Student

Discente:

* Gabriel D'assumpção de Carvalho

Data: 30/07/2024

Introdução

O objetivo deste trabalho é analisar a distribuição dos retornos do Ibovespa. Para calcular os retornos, utilizaremos o preço de fechamento ajustado diário no período de 27 de abril de 2020 a 30 de julho de 2024. Os dados serão obtidos a partir do Yahoo Finance, e você pode acessá-los diretamente aqui.

Neste estudo, serão plotados histogramas dos retornos diários do Ibovespa para verificar graficamente qual distribuição de probabilidade melhor se ajusta aos dados. Com base na análise gráfica, será possível gerar valores aleatórios dos retornos para simulações futuras. Além disso, realizaremos três testes de Kolmogorov-Smirnov (K-S) para dados contínuos, com as seguintes hipóteses nulas:

1. H0: Retorno segue uma distribuição Normal;
2. H0: Retorno segue uma distribuição T-studante com (n - 1) graus de liberdade;
3. H0: Retorno segue uma distribuição T-studante com (n - 2) graus de liberdade.

Metodologia

Os dados foram obtidos através da biblioteca `yfinance`, focando nos preços de fechamento ajustados do índice Ibovespa. Após a obtenção dos dados, foram calculados os retornos diários, que foram então ordenados para posterior análise.

Análise Gráfica

Foi plotado o histograma dos retornos diários do Ibovespa com o objetivo de identificar visualmente a distribuição dos dados. Além disso, geramos valores aleatórios dos retornos para comparação com a distribuição empírica observada.

Geração de Valores

Após a análise exploratória dos dados, serão gerados valores aleatórios dos retornos utilizando a média e variância amostral dos retornos.

Testes de Kolmogorov-Smirnov

Foram realizados três testes de Kolmogorov-Smirnov para avaliar se os retornos diários seguem uma distribuição Normal, uma distribuição t-Student com (n - 1) graus de liberdade, e uma distribuição t-Student com (n - 2) graus de liberdade. Estes testes permitem verificar a aderência das distribuições teóricas aos dados empíricos.

Resultados Esperados

Através desta análise, esperamos identificar a distribuição que melhor descreve os retornos do Ibovespa, o que poderá auxiliar na modelagem de previsões e na tomada de decisões de investimento.

Biblioteca utilizadas

1. yfinance: Para obtenção dos dados financeiros.
2. numpy: Para cálculos estatísticos.
3. matplotlib e seaborn: Para criação de gráficos.
4. scipy: Para realização dos testes estatísticos.
5. warnings: Para remoção de avisos.

In [ ]:
```
import yfinance as yf
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
from scipy.stats import t, norm, uniform, mode
```

In [ ]:
```
# Desativa todos os avisos
warnings.filterwarnings("ignore")
```

Obtenção dos Dados

In [ ]:
```
# Obtendo os dados na COGN3 pelo Yahoo Finance
data = yf.download("^BVSP", start="2020-04-27", end="2024-07-30")['Adj Close
```

```
[**********100%%**********]  1 of 1 completed
```

Cálculo dos Retornos

In [ ]:
```
# Criando uma variável que contêm os retornos
ret = data.pct_change().dropna()

ret = ret

# Ordenar os retornos
ret_sort = np.sort(ret)
```

In [ ]:
```
ret.describe()
```

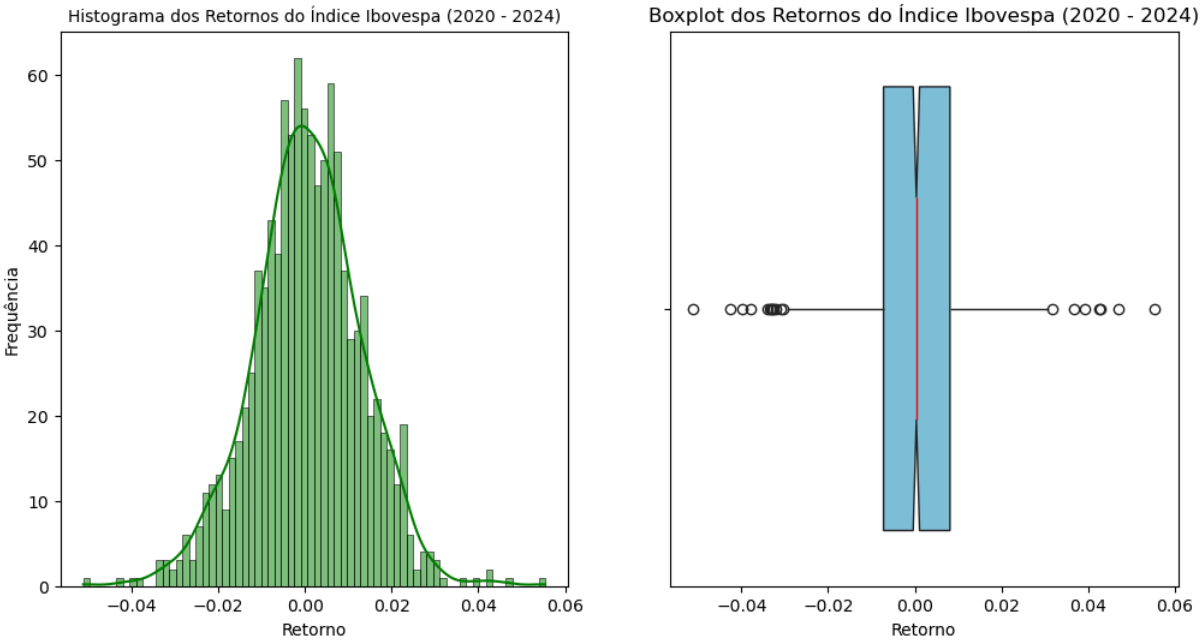
Out[ ]:
```
count    1059.000000
mean        0.000535
std         0.012501
min        -0.051201
25%        -0.007215
50%         0.000363
75%         0.008116
max         0.055409
Name: Adj Close, dtype: float64
```

Na análise estatística dos retornos do Ibovespa, observamos que, em média, o índice apresenta um retorno de 0,000535, com um desvio padrão de 0,012501. Isso indica que a maioria dos retornos está no intervalo de -0,011966 a 0,013036. Além disso, os retornos variam de um mínimo de -0,51201 a um máximo de 0,55409.

A proximidade entre a média (0,000535) e a mediana (0,000363) sugere uma simetria nos dados. No entanto, para avaliar melhor a distribuição e a simetria dos retornos, é necessário examinar o histograma e o boxplot. Esses gráficos proporcionarão uma visão mais clara e detalhada da distribuição dos dados.

In [ ]:
```
# Criação do grafico histograma e boxplot dos retornos
plt.figure(figsize=(12,6))
plt.subplot(1, 2, 1)
sns.histplot(ret, kde=True, bins=70 ,color='green')
plt.title("Histograma dos Retornos do Índice Ibovespa (2020 - 2024)", fontsi
plt.xlabel("Retorno")
plt.ylabel("Frequência")
```

```
plt.subplot(1, 2, 2)
sns.boxplot(ret, orient='h', color='green', notch=True, showcaps=False,
boxprops={"facecolor": (0, .5, .7, .5)},
medianprops={"color": "r", "linewidth": 1})
plt.title("Boxplot dos Retornos do Índice Ibovespa (2020 - 2024)")
plt.xlabel("Retorno")
plt.show()
```



Podemos observar no histograma que os retornos diários do Ibovespa apresentam uma forma que se aproxima de uma distribuição normal, porém com caudas ligeiramente mais pesadas. Isso também é evidenciado pelos outliers observados no boxplot. Esses outliers indicam que os retornos extremos, tanto positivos quanto negativos, ocorrem com maior frequência do que o esperado em uma distribuição normal.

Essa característica sugere que a distribuição dos retornos do Ibovespa pode ser melhor modelada por uma distribuição t de Student, que é conhecida por capturar melhor a frequência de eventos extremos devido às suas caudas mais pesadas.

Com base nesta observação, vamos gerar empiricamente a função de probabilidade acumulada (ECDF) dos retornos. Em seguida, geraremos valores aleatórios considerando que os retornos seguem uma distribuição normal, utilizando a média e o desvio padrão amostral dos dados observados. Isso nos permitirá comparar a distribuição empírica dos retornos com a distribuição teórica normal e avaliar a adequação de cada modelo.

In [ ]:
```
# Número de amostras desejadas
n = len(ret)

# Médidas dos retornos
media_ret = ret.mean()
dp_ret = ret.std()
```

Geração de Valores Aleatórios

No código abaixo, estamos gerando uma amostra aleatória com a mesma quantidade de observações dos retornos, totalizando 1059 observações. Assumimos que (X) segue uma distribuição normal com média $\mu = 0.00053$ e variância $\sigma^2 = 0.0125^2$.

In [ ]:
```
# Gerando valores aleatorios
X = []
for i in range(n):
  X.append((dp_ret * np.sqrt(12 / 100000) * (np.random.uniform(0, 1, 100000)
X = np.array(X)
X_sort = np.sort(X)
```

In [ ]:
```
print("μ_{ret} = ", media_ret, "\nσ_{ret} = ", dp_ret, '\n')
print("μ_{X} = ", X.mean(), "\nσ_{X} = ", X.std())
```

```
μ_{ret} =  0.0005352541442946981
σ_{ret} =  0.012500951090858341

μ_{X} =  -0.00023626080651362354
σ_{X} =  0.012418098550885668
```

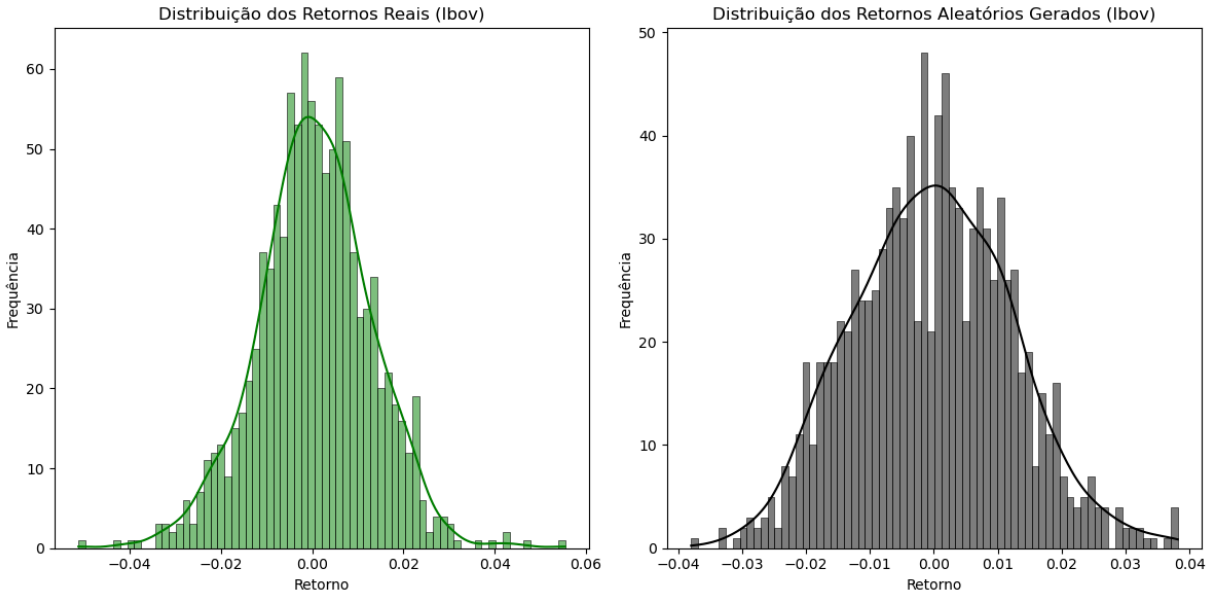
Podemos ver acima que tanto a média quanto o desvio padrão dos retornos são bastante próximos dos valores da amostra aleatória.

In [ ]:
```
# Plotando os histogramas para comparação
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
sns.histplot(ret, kde=True, bins=70, color='green')
plt.title('Distribuição dos Retornos Reais (Ibov)')
plt.xlabel('Retorno')
plt.ylabel('Frequência')

plt.subplot(1, 2, 2)
sns.histplot(X, kde=True, bins=70, color='black')
plt.title('Distribuição dos Retornos Aleatórios Gerados (Ibov)')
plt.xlabel('Retorno')
plt.ylabel('Frequência')

plt.tight_layout()
plt.show()
```



No histograma acima, em verde estão os retornos reais do Ibovespa, enquanto em preto estão os retornos aleatórios, assumindo que os dados seguem uma distribuição normal. Podemos visualizar que os retornos gerados têm um comportamento muito próximo aos retornos reais.

Para verificar melhor a semelhança, vamos calcular a função empírica de probabilidade acumulada (ECDF) das duas distribuições. Em seguida, realizaremos o teste estatístico de Kolmogorov-Smirnov para avaliar a adequação das distribuições teórica e empírica.

Função Empírica de Probabilidade Acumulada

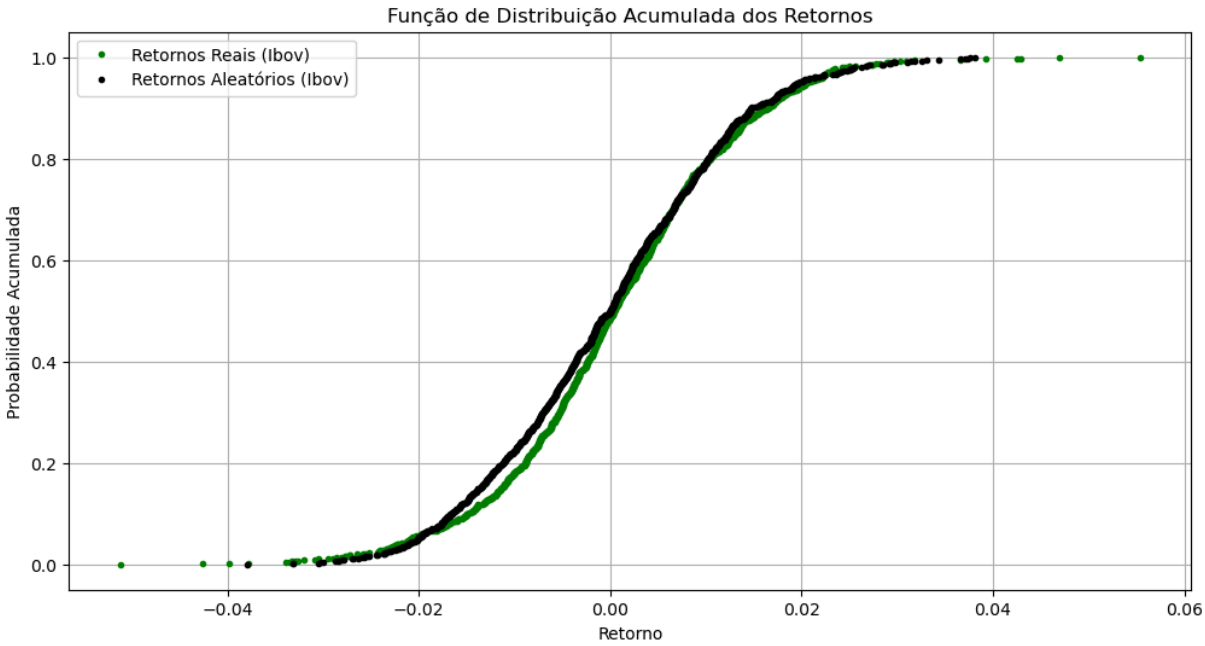
In [ ]:
```
# Calcular a ECDF manualmente
Fn = np.arange(1, n + 1) / n
```

Para gerar a ECDF (Função de Distribuição Acumulada) dos retornos reais, assumimos que a probabilidade de cada valor é de (\frac{1}{n}). Assim, a probabilidade acumulada é dada por:

- $\text{ret}_1 = \frac{1}{n}$
- $\text{ret}_2 = \frac{2}{n}$
- $\text{ret}_n = 1$

In [ ]:
```
# Plotar a ECDF
plt.figure(figsize=(12, 6))
plt.plot(ret_sort, Fn, color='green' ,marker='.', linestyle='none', label='R
plt.plot(X_sort, Fn, color='black' ,marker='.', linestyle='none', label='Ret
plt.title('Função de Distribuição Acumulada dos Retornos')
plt.xlabel('Retorno')
plt.ylabel('Probabilidade Acumulada')
plt.grid(True)
```

```
plt.legend()
plt.show()
```



No gráfico de dispersão da função de distribuição acumulada (ECDF) dos retornos reais e dos gerados, podemos observar que as curvas se aproximam bastante. Isso sugere que os dados se ajustam bem a uma distribuição normal.

Para confirmar essa observação, realizaremos o teste estatístico de Kolmogorov-Smirnov (K-S), que nos permitirá avaliar se há uma diferença estatisticamente significativa entre as duas distribuições.

Testes de Kolmogorov-Smirnov para Distribuição Normal

No código abaixo, realizaremos o teste de Kolmogorov-Smirnov com 95% de confiança, realizando 10.000 simulações para verificar a probabilidade de que os valores reais se aproximam de uma distribuição normal.

In [ ]:
```
dados_norm = np.sort(np.random.normal(0, 1, n))
Fn_norm = norm.cdf(dados_norm)
```

In [ ]:
```
D1 = np.zeros(n)
D2 = np.zeros(n)
```

In [ ]:
```
# Preencher D_1 e D_2
for i in range(n):
    D1[i] = 1 / n - Fn_norm[i]
    D2[i] = Fn_norm[i] - (i - 1) / n
```

```
# Calcular D_calc
D_calc = np.max(np.maximum(D1, D2))
```

In [ ]:
```
# Simulação do p-valor
alpha = 0.05  # nível de significância
n_rep = 10000
```

In [ ]:
```
D1 = np.zeros(n)
D2 = np.zeros(n)
```

In [ ]:
```
contador = 0
for _ in range(n_rep):
    aux = np.sort(np.random.uniform(0, 1, n))
    for j in range(len(aux)):
        D1[j] = j/n - uniform.cdf(aux[j])
        D2[j] = uniform.cdf(aux[j]) - (j - 1) / n
    D = np.max(np.maximum(D1, D2))
    if D >= D_calc:
        contador += 1

pvalor = contador / n_rep

if pvalor < alpha:
    print("Rejeite H_0")
else:
    print("Não rejeite H_0")

print(f'Estatística do teste D: {D_calc}')
print(f'Valor-p: {pvalor}')
```

```
Não rejeite H_0
Estatística do teste D: 0.029083710032657084
Valor-p: 0.3319
```

O teste de Kolmogorov-Smirnov foi aplicado para avaliar se os valores reais seguem uma distribuição normal. A estatística do teste, D, é 0.0293, que representa a maior diferença absoluta entre a função de distribuição acumulada empírica (ECDF) dos dados reais e a função de distribuição acumulada teórica da normal.

O valor-p obtido é 0.3319. Este valor-p indica a probabilidade de observar uma estatística de teste tão extrema quanto a observada, assumindo que a hipótese nula H_0 (de que os dados seguem uma distribuição normal) é verdadeira.

Como o valor-p é significativamente maior do que o nível de significância comum de 0.05, não rejeitamos a hipótese nula. Isso sugere que não há evidências suficientes para concluir que os dados reais diferem da distribuição normal. Portanto, com base neste teste, os dados podem ser considerados como seguindo uma distribuição normal, de acordo com o critério estabelecido.

Testes de Kolmogorov-Smirnov para T-studante com (n - 1) gl

No código abaixo, realizaremos o teste de Kolmogorov-Smirnov com 95% de confiança, realizando 10.000 simulações para verificar a probabilidade de que os valores reais se aproximam da distribuição t de studante com $(n-1)$ grau de liberdade.

In [ ]:
```
dados_ts1 = np.sort(t.rvs(n - 1, size=n))
Fn_ts1 = t.cdf(dados_ts1, n - 1)
```

In [ ]:
```
D1 = np.zeros(n)
D2 = np.zeros(n)
```

In [ ]:
```
# Preencher D_1 e D_2
for i in range(n):
    D1[i] = 1 / n - Fn_norm[i]
    D2[i] = Fn_norm[i] - (i - 1) / n

# Calcular D_calc
D_calc = np.max(np.maximum(D1, D2))
```

In [ ]:
```
D1 = np.zeros(n)
D2 = np.zeros(n)
```

In [ ]:
```
contador = 0
for _ in range(n_rep):
    aux = np.sort(np.random.uniform(0, 1, n))
    for j in range(len(aux)):
        D1[j] = j/n - uniform.cdf(aux[j])
        D2[j] = uniform.cdf(aux[j]) - (j - 1) / n
    D = np.max(np.maximum(D1, D2))
    if D >= D_calc:
        contador += 1

pvalor = contador / n_rep

if pvalor < alpha:
    print("Rejeite H_0")
else:
    print("Não rejeite H_0")

print(f'Estatística do teste D: {D_calc}')
print(f'Valor-p: {pvalor}')
```

```
Não rejeite H_0
Estatística do teste D: 0.029083710032657084
Valor-p: 0.3234
```

Com base nos resultados do teste de Kolmogorov-Smirnov acima, não rejeitamos a hipótese nula (H_0). A grande quantidade de observações na amostra faz com que a distribuição t de Student com $(n-1)$ graus de liberdade se aproxime muito da distribuição normal. Esta aproximação é evidente no teste de Kolmogorov-Smirnov, que avalia a adequação dos dados à distribuição normal.

O valor-p de 0.3234 sugere que a diferença entre a distribuição dos dados reais e a distribuição t de studant é muito pequena. Além disso, a estatística (D) do teste permaneceu a mesma, indicando que a discrepância entre as distribuições não é significativa. Portanto, os dados podem ser considerados como seguindo uma distribuição t de studant, de acordo com o teste realizado.

Testes de Kolmogorov-Smirnov para T-studante com (n - 2) gl

No código abaixo, realizaremos o teste de Kolmogorov-Smirnov com 95% de confiança, realizando 10.000 simulações para verificar a probabilidade de que os valores reais se aproximam da distribuição t de studante com $(n-2)$ grau de liberdade.

In [ ]:
```
dados_ts2 = np.sort(t.rvs(n - 2, size=n))
Fn_ts2 = t.cdf(dados_ts2, n - 2)
```

In [ ]:
```
D1 = np.zeros(n)
D2 = np.zeros(n)
```

In [ ]:
```
# Preencher D_1 e D_2
for i in range(n):
    D1[i] = 1 / n - Fn_norm[i]
    D2[i] = Fn_norm[i] - (i - 1) / n

# Calcular D_calc
D_calc = np.max(np.maximum(D1, D2))
```

In [ ]:
```
D1 = np.zeros(n)
D2 = np.zeros(n)
```

In [ ]:
```
contador = 0
for _ in range(n_rep):
    aux = np.sort(np.random.uniform(0, 1, n))
    for j in range(len(aux)):
        D1[j] = j/n - uniform.cdf(aux[j])
        D2[j] = uniform.cdf(aux[j]) - (j - 1) / n
    D = np.max(np.maximum(D1, D2))
    if D >= D_calc:
        contador += 1

pvalor = contador / n_rep

if pvalor < alpha:
    print("Rejeite H_0")
else:
    print("Não rejeite H_0")

print(f'Estatística do teste D: {D_calc}')
print(f'Valor-p: {pvalor}')
```

```
Não rejeite H_0
Estatística do teste D: 0.029083710032657084
Valor-p: 0.3315
```

Os resultados indicam que não rejeitamos a hipótese nula (H_0). Isso sugere que, com um nível de confiança de 95%, os retornos diários do índice Ibovespa podem ser bem representados por uma distribuição t de Student com $(n-2)$ graus de liberdade.

O valor-p de 0.3315 e a estatística (D) baixa indicam que a discrepância entre os dados reais e a distribuição t é insignificante. Portanto, além de poderem ser descritos por uma distribuição normal, os dados também se ajustam adequadamente a uma distribuição t com $(n-2)$ graus de liberdade.

Conclusão

Com base neste estudo, podemos afirmar com 95% de confiança que os retornos diários do índice Ibovespa podem ser modelados utilizando a distribuição normal ou as distribuições t de Student com $(n-1)$ ou $(n-2)$ graus de liberdade.

No entanto, para a geração de valores aleatórios, optou-se por utilizar uma distribuição normal devido à sua simplicidade na geração da amostra. Em contraste, no setor financeiro, é comum modelar dados de retornos com uma distribuição t de Student, pois esta distribuição melhor captura a ocorrência de eventos extremos, como grandes variações de preço em um único dia, que são frequentes em dados financeiros.

O intuito de identificar a distribuição do índice Ibovespa tem como objetivo ajudar na criação de reservas financeiras, onde é preciso criar intervalos de confiança para o ganho e a perda da aplicação. Para realizar esses testes estatísticos, é fundamental identificar qual distribuição a variável de interesse segue.