CÓDIGOS

terça-feira, 19 de setembro de 2023 21:20

Identificando espaços amostrais: A moeda e o dado:

```python
import itertools
# Resultados para o lançamento da moeda e do dado
moeda = ['Cara', 'Coroa']
dado = ['1', '2', '3', '4', '5', '6']
# Criação do espaço amostral
omega = [resultado for resultado in itertools.product(moeda, dado)]
# Visualização dos elementos do espaço amostral
print('** Espaço amostral **')
for resultado in omega:
  print(resultado)
# Número de elementos no espaço amostral
n_omega = len(omega)
print(f'\nExistem {n_omega} elementos no espaço amostral.')
```

# Resultados para o lançamento da moeda e do dado
moeda = ['Cara', 'Coroa']
dado = ['1', '2', '3', '4', '5', '6']

- Neste trecho, você está definindo duas listas, `moeda` e `dado`, que representam os possíveis resultados de um lançamento de moeda e de um lançamento de dado, respectivamente. `moeda` contém dois resultados possíveis (Cara e Coroa), enquanto `dado` contém seis resultados possíveis (os números de 1 a 6).

# Criação do espaço amostral
omega = [resultado for resultado in itertools.product(moeda, dado)]

- Aqui, você está criando o espaço amostral chamado `omega`. Você usa a função `itertools.product()` para criar todas as combinações possíveis entre os resultados de `moeda` e `dado`. Isso resulta em uma lista de tuplas que representam todos os resultados possíveis quando você lança a moeda e o dado.

# Visualização dos elementos do espaço amostral
print('** Espaço amostral **')
for resultado in omega:
  print(resultado)

- Este trecho exibe os elementos do espaço amostral. Ele itera sobre cada tupla em `omega` e imprime cada resultado no formato ('Cara', '1'), ('Cara', '2'), ('Cara', '3'), ..., ('Coroa', '6').

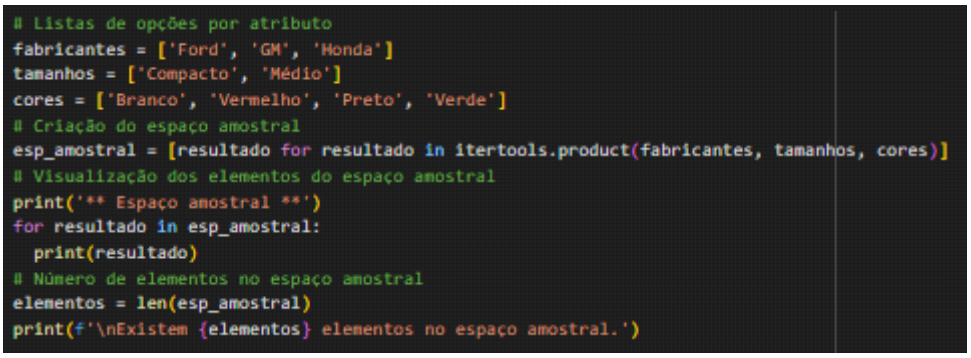
# Número de elementos no espaço amostral
n_omega = len(omega)
print(f'\nExistem {n_omega} elementos no espaço amostral.')

- Por fim, você calcula o número de elementos no espaço amostral `omega` usando a função

`len(omega)` e imprime o resultado. Isso informa quantas combinações únicas existem no espaço amostral. No seu caso, o número de elementos no espaço amostral será 2 (moeda) multiplicado por 6 (dado), resultando em 12 elementos únicos.

O resultado é a criação e visualização do espaço amostral que representa todos os resultados possíveis quando você lança uma moeda e um dado. Esse espaço amostral é fundamental em teoria da probabilidade para entender e calcular probabilidades de eventos.

Identificando espaços amostrais: a compra do carro novo:



# Listas de opções por atributo
fabricantes = ['Ford', 'GM', 'Honda']
tamanhos = ['Compacto', 'Médio']
cores = ['Branco', 'Vermelho', 'Preto', 'Verde']

- Neste trecho, você está definindo três listas, `fabricantes`, `tamanhos` e `cores`, que representam as opções possíveis para três atributos diferentes:
  - `fabricantes`: Contém as opções de fabricantes de carros (Ford, GM e Honda).
  - `tamanhos`: Contém as opções de tamanhos de carros (Compacto e Médio).
  - `cores`: Contém as opções de cores de carros (Branco, Vermelho, Preto e Verde).

# Criação do espaço amostral
esp_amostral = [resultado for resultado in itertools.product(fabricantes, tamanhos, cores)]

- Aqui, você está criando o espaço amostral chamado `esp_amostral`. Você usa a função `itertools.product()` para criar todas as combinações possíveis entre as opções de fabricantes, tamanhos e cores. Isso resulta em uma lista de tuplas que representam todas as combinações possíveis de atributos para um carro.

# Visualização dos elementos do espaço amostral
print('** Espaço amostral **')
for resultado in esp_amostral:
  print(resultado)

- Este trecho exibe os elementos do espaço amostral. Ele itera sobre cada tupla em `esp_amostral` e imprime cada resultado no formato ('Ford', 'Compacto', 'Branco'), ('Ford', 'Compacto', 'Vermelho'), ..., ('Honda', 'Médio', 'Verde').

```
# Número de elementos no espaço amostral
elementos = len(esp_amostral)
print(f'\nExistem {elementos} elementos no espaço amostral.')
```

- Por fim, você calcula o número de elementos no espaço amostral `esp_amostral` usando a função `len(esp_amostral)` e imprime o resultado. Isso informa quantas combinações únicas existem no espaço amostral, ou seja, quantos tipos diferentes de carros podem ser criados com as opções de fabricantes, tamanhos e cores fornecidas.

O resultado é a criação e visualização do espaço amostral que representa todas as combinações possíveis de fabricantes, tamanhos e cores para carros. Esse espaço amostral pode ser útil em diversos contextos, como modelagem de produtos, planejamento de produção ou análise de preferências de clientes.

Cálculo de probabilidades clássicas: o carro novo:

```
# Probabilidade de que o carro novo seja um Ford
evento_1 = [escolha for escolha in esp_amostral if 'Ford' in escolha]
print('** Evento 1: Carro novo é um Ford **')
for resultado in evento_1:
  print(resultado)
prob_1 = len(evento_1)/len(esp_amostral)
print(f'Probabilidade do evento 1 = {prob_1*100:.2f}%\n')
# Probabilidade de que o carro novo seja da cor verde
evento_2 = [escolha for escolha in esp_amostral if 'Verde' in escolha]
print('** Evento 2: Carro novo é da cor verde **')
for resultado in evento_2:
  print(resultado)
prob_2 = len(evento_2)/len(esp_amostral)
print(f'Probabilidade do evento 2 = {prob_2*100:.2f}%\n')
# Probabilidade de que o carro novo seja um compacto
evento_3 = [escolha for escolha in esp_amostral if 'Compacto' in escolha]
print('** Evento 3: Carro novo é um compacto **')
for resultado in evento_3:
  print(resultado)
prob_3 = len(evento_3)/len(esp_amostral)
print(f'Probabilidade do evento 3 = {prob_3*100:.2f}%\n')
# Probabilidade de que o carro novo seja um compacto Honda
evento_4 = [escolha for escolha in esp_amostral if 'Honda' in escolha if 'Compacto' in escolha]
print('** Evento 4: Carro novo é um compacto Honda **')
for resultado in evento_4:
  print(resultado)
prob_4 = len(evento_4)/len(esp_amostral)
print(f'Probabilidade do evento 4 = {prob_4*100:.2f}%\n')
# Probabilidade de que o carro novo um GM vermelho
evento_5 = [escolha for escolha in esp_amostral if 'GM' in escolha if 'Vermelho' in escolha]
print('** Evento 5: Carro novo é um GM vermelho **')
for resultado in evento_5:
  print(resultado)
prob_5 = len(evento_5)/len(esp_amostral)
print(f'Probabilidade do evento 5 = {prob_5*100:.2f}%\n')
```

```
# Probabilidade de que o carro novo seja um Ford
evento_1 = [escolha for escolha in esp_amostral if 'Ford' in escolha]
print('** Evento 1: Carro novo é um Ford **')
for resultado in evento_1:
  print(resultado)
prob_1 = len(evento_1)/len(esp_amostral)
print(f'Probabilidade do evento 1 = {prob_1*100:.2f}%\n')
```

- Neste trecho, você está calculando a probabilidade de que o carro novo seja um Ford. Para fazer isso, você cria uma lista chamada `evento_1` que contém todas as escolhas (combinações de

fabricante, tamanho e cor) em `esp_amostral` que incluem a palavra 'Ford'. Em seguida, você imprime os resultados dessas escolhas para visualização e calcula a probabilidade dividindo o número de escolhas que satisfazem o evento pelo número total de escolhas em `esp_amostral`.

```
# Probabilidade de que o carro novo seja da cor verde
evento_2 = [escolha for escolha in esp_amostral if 'Verde' in escolha]
print('** Evento 2: Carro novo é da cor verde **')
for resultado in evento_2:
  print(resultado)
prob_2 = len(evento_2)/len(esp_amostral)
print(f'Probabilidade do evento 2 = {prob_2*100:.2f}%\n')
```

- Aqui, você está fazendo o mesmo processo, mas agora está calculando a probabilidade de que o carro novo seja da cor verde. Mais uma vez, você cria a lista `evento_2` que contém todas as escolhas que incluem a palavra 'Verde', imprime os resultados e calcula a probabilidade.

Os trechos seguintes, `evento_3`, `evento_4` e `evento_5`, seguem o mesmo padrão, calculando probabilidades para diferentes eventos:

- `evento_3` calcula a probabilidade de que o carro novo seja um compacto.
- `evento_4` calcula a probabilidade de que o carro novo seja um compacto da marca Honda.
- `evento_5` calcula a probabilidade de que o carro novo seja da marca GM e de cor vermelha.

Em todos os casos, você está filtrando as escolhas que atendem aos critérios do evento, contando essas escolhas e calculando a probabilidade em relação ao espaço amostral total. As probabilidades calculadas são exibidas como porcentagens no final de cada evento.

Cálculo de probabilidades empíricas:

```
import pandas as pd
# DataFrame com os dados da pesquisa
pesquisa = pd.DataFrame({'Frequência': [156, 312, 254, 195, 58]})
pesquisa.index = ['18 a 22', '23 a 35', '36 a 49', '50 a 65', 'acima de 65']
display(pesquisa)
# Cálculo das probabilidades
pesquisa['Probabilidades'] = pesquisa['Frequência']/pesquisa['Frequência'].sum()*100
display(pesquisa)
```

```
# DataFrame com os dados da pesquisa
pesquisa = pd.DataFrame({'Frequência': [156, 312, 254, 195, 58]})
```

- Neste trecho, você está criando um DataFrame chamado `pesquisa` com uma coluna chamada 'Frequência'. Esse DataFrame representa os resultados de uma pesquisa, onde cada valor na coluna 'Frequência' representa o número de pessoas em diferentes faixas etárias.

```
pesquisa.index = ['18 a 22', '23 a 35', '36 a 49', '50 a 65', 'acima de 65']
```

- Aqui, você está definindo os rótulos de índice para o DataFrame `pesquisa`. Esses rótulos representam as faixas etárias correspondentes às frequências de pessoas na pesquisa. Cada rótulo de índice corresponde à mesma ordem das frequências no DataFrame.

Cálculo das probabilidades
pesquisa['Probabilidades'] = pesquisa['Frequência']/pesquisa['Frequência'].sum()*100

- Neste trecho, você está calculando as probabilidades correspondentes a cada faixa etária. Você divide o valor da coluna 'Frequência' pelo total de frequências no DataFrame (`pesquisa['Frequência'].sum()`) e multiplica por 100 para expressar as probabilidades em porcentagem. Essas probabilidades são armazenadas em uma nova coluna chamada 'Probabilidades' no DataFrame `pesquisa`.

display(pesquisa)

- Por fim, você exibe o DataFrame `pesquisa` após o cálculo das probabilidades. Isso mostrará as frequências e as probabilidades associadas a cada faixa etária na pesquisa.

O resultado final é um DataFrame que inclui as frequências e as probabilidades das diferentes faixas etárias na pesquisa, facilitando a análise dos resultados em termos de distribuição por faixa etária.

Mais probabilidades binomiais:

```
import random
import matplotlib.pyplot as plt

# Função para lançamentos da moeda
def moeda(lancamentos):
  cara = 0
  for _ in range(lancamentos):
    if random.random() <= 0.5:
        cara +=1
  return cara/lancamentos

# Função para geração de cada experimento
def experimentos(lista_lancamentos):
    proporcoes = []
    for n_lancamento in lista_lancamentos:
        proporcoes.append(moeda(n_lancamento))
    return proporcoes

# Simulação
testes = list(range(1,5000,50))
resultados = experimentos(testes)
# Representação gráfica
plt.scatter(testes, resultados)
plt.title('Probaillidade de se obter "cara"')
plt.xlabel('Número de lançamentos')
plt.ylabel('Proporção de caras')
plt.grid()
```

# Função para lançamentos da moeda
def moeda(lancamentos):
  cara = 0
  for _ in range(lancamentos):
    if random.random() <= 0.5:
        cara +=1
  return cara/lancamentos

- Neste trecho, você define uma função chamada `moeda` que simula lançamentos de uma moeda. A função aceita um argumento `lancamentos`, que representa o número de lançamentos a serem

realizados. A função começa com uma contagem de `cara` igual a zero. Em seguida, ela itera `lancamentos` vezes, gerando números aleatórios entre 0 e 1 (usando `random.random()`) e verifica se o número gerado é menor ou igual a 0,5 (representando "cara"). Se for, incrementa o contador `cara`. No final, a função retorna a proporção de "caras" em relação ao número total de lançamentos.

python
```
# Função para geração de cada experimento
def experimentos(lista_lancamentos):
   proporcoes = []
   for n_lancamento in lista_lancamentos:
      proporcoes.append(moeda(n_lancamento))
   return proporcoes
```

- Aqui, você define outra função chamada `experimentos` que realiza uma série de experimentos para diferentes números de lançamentos. Ela aceita uma lista chamada `lista_lancamentos`, que contém os números de lançamentos que você deseja testar. A função itera sobre essa lista, chamando a função `moeda` para cada número de lançamentos e armazena as proporções de "caras" resultantes na lista `proporcoes`.

python
```
# Simulação
testes = list(range(1,5000,50))
resultados = experimentos(testes)
```

- Neste trecho, você define a lista `testes` que contém uma sequência de números de 1 a 5000 (inclusive), com incrementos de 50. Esses números representam o número de lançamentos que você deseja testar. Em seguida, você chama a função `experimentos` com a lista `testes` e armazena os resultados na lista `resultados`. Isso simula a probabilidade de se obter "cara" para diferentes números de lançamentos.

python
```
# Representação gráfica
plt.scatter(testes, resultados)
plt.title('Probabilidade de se obter "cara"')
plt.xlabel('Número de lançamentos')
plt.ylabel('Proporção de caras')
plt.grid()
```

- Finalmente, você cria um gráfico de dispersão usando `plt.scatter` para representar os resultados da simulação. No eixo x, você coloca o número de lançamentos (`testes`), e no eixo y, você coloca a proporção de "caras" correspondente (`resultados`). Você adiciona um título ao gráfico, rótulos aos eixos e uma grade para facilitar a leitura do gráfico. Esse gráfico mostra visualmente como a probabilidade de obter "cara" varia com o número de lançamentos da moeda.