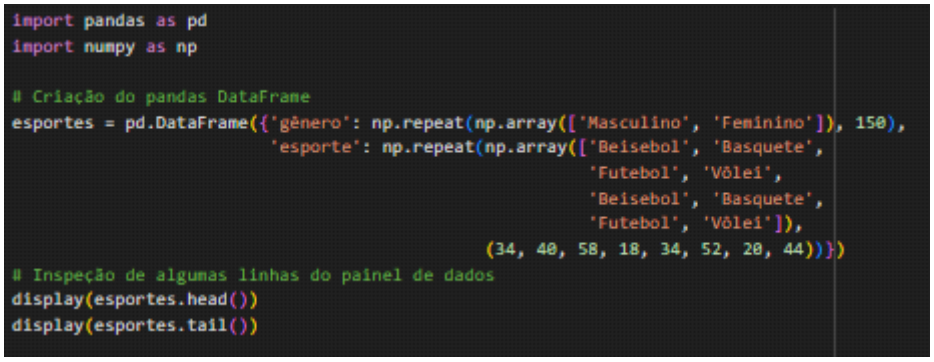
CÓDIGOS

terça-feira, 19 de setembro de 2023 21:33

Probabilidades condicionais - Esportes:



```
import pandas as pd
import numpy as np

# Criação do pandas DataFrame
esportes = pd.DataFrame({'gênero': np.repeat(np.array(['Masculino', 'Feminino']), 150),
                         'esporte': np.repeat(np.array(['Beisebol', 'Basquete',
                                                        'Futebol', 'Vôlei',
                                                        'Beisebol', 'Basquete',
                                                        'Futebol', 'Vôlei']),
                                              (34, 40, 58, 18, 34, 52, 20, 44))})
# Inspeção de algumas linhas do painel de dados
display(esportes.head())
display(esportes.tail())
```

Este código em Python utiliza a biblioteca Pandas e a biblioteca NumPy para criar um DataFrame chamado `esportes` que contém informações sobre gênero e esportes. Vou explicar cada trecho do código:

```
# Importação das bibliotecas Pandas e NumPy
import pandas as pd
import numpy as np
```

Neste trecho, você importa as bibliotecas necessárias, Pandas e NumPy, para lidar com manipulação de dados e operações numéricas.

```
# Criação do pandas DataFrame
esportes = pd.DataFrame({'gênero': np.repeat(np.array(['Masculino', 'Feminino']), 150),
                         'esporte': np.repeat(np.array(['Beisebol', 'Basquete',
                                                        'Futebol', 'Vôlei',
                                                        'Beisebol', 'Basquete',
                                                        'Futebol', 'Vôlei']),
                                              (34, 40, 58, 18, 34, 52, 20, 44))})
```

Neste trecho, você cria o DataFrame `esportes`. Aqui está o que acontece:

- Você usa o construtor `pd.DataFrame()` para criar um novo DataFrame.
- O dicionário passado como argumento para o construtor contém duas colunas: 'gênero' e 'esporte'.
- No campo 'gênero', você utiliza a função `np.repeat` do NumPy para repetir as strings 'Masculino' e 'Feminino' 150 vezes cada. Isso cria uma coluna que alterna entre 'Masculino' e 'Feminino' ao longo das linhas.
- No campo 'esporte', você utiliza a função `np.repeat` novamente para criar uma coluna que contém os esportes 'Beisebol', 'Basquete', 'Futebol' e 'Vôlei' com as respectivas quantidades de repetições especificadas em uma tupla (34, 40, 58, 18, 34, 52, 20, 44).

```
# Inspeção de algumas linhas do painel de dados
display(esportes.head())
display(esportes.tail())
```

Nesses trechos, você utiliza a função `display()` para mostrar as primeiras (com `head()`) e últimas (com `tail()`) linhas do DataFrame `esportes`. Isso é útil para verificar o conteúdo do DataFrame e garantir que ele foi criado corretamente.

Tabela de distribuição conjunta de frequências:

```
# Construção da tabela de distrbuição conjunta
dist_conj = pd.crosstab(index=esportes['gênero'],
                        columns=esportes['esporte'], margins=True)
# Inspeção da tabela de distribuição conjunta
display(dist_conj)
```

Neste trecho de código, você está construindo uma tabela de distribuição conjunta com base no DataFrame `esportes`. Vou explicar cada parte do código:

```
# Construção da tabela de distribuição conjunta
dist_conj = pd.crosstab(index=esportes['gênero'],
                        columns=esportes['esporte'], margins=True)
```

1. `pd.crosstab()`: Você está usando a função `crosstab` do Pandas para criar uma tabela de distribuição conjunta. Esta função é útil para resumir e contar a frequência de ocorrência de combinações entre duas ou mais variáveis.

2. `index=esportes['gênero']`: Você especifica que as categorias da coluna 'gênero' do DataFrame `esportes` serão usadas como o índice da tabela de distribuição conjunta.

3. `columns=esportes['esporte']`: Você especifica que as categorias da coluna 'esporte' do DataFrame `esportes` serão usadas como as colunas da tabela de distribuição conjunta.

4. `margins=True`: Ao definir `margins` como `True`, você solicita que a tabela de distribuição conjunta inclua margens, ou seja, totais parciais nas linhas e colunas.

```
# Inspeção da tabela de distribuição conjunta
display(dist_conj)
```

Nesse trecho, você usa a função `display()` para mostrar a tabela de distribuição conjunta resultante, que foi armazenada na variável `dist_conj`. A tabela exibirá a frequência de ocorrência de cada combinação entre gênero e esporte, bem como totais parciais nas margens. Essa tabela é útil para entender a distribuição conjunta das variáveis 'gênero' e 'esporte' em seu conjunto de dados.

Cálculo de probabilidades condicionais:

```
# Probabilidade de ser do gênero masculino, dado que é fã de beisebol
prob = dist_conj.iloc[1, 1] / dist_conj.iloc[2, 1]
print(f'P(Masculino|Beisebol) = {prob*100:.2f}%')

# Probabilidade de ser fã de basquete, dado que é do gênero feminino
prob = dist_conj.iloc[0, 0] / dist_conj.iloc[0, 4]
print(f'P(Basquete|Feminino) = {prob*100:.2f}%')
```

Neste trecho de código, você está calculando probabilidades condicionais com base na tabela de distribuição conjunta `dist_conj`. Vou explicar cada cálculo:

1. *Probabilidade de ser do gênero masculino, dado que é fã de beisebol:*

```
prob = dist_conj.iloc[1, 1] / dist_conj.iloc[2, 1]
```

- `dist_conj.iloc[1, 1]`: Isso obtém a contagem de pessoas que são do gênero masculino e fãs de beisebol na tabela de distribuição conjunta.
- `dist_conj.iloc[2, 1]`: Isso obtém a contagem total de pessoas que são fãs de beisebol na tabela de distribuição conjunta.

A probabilidade condicional P(Masculino|Beisebol) é calculada dividindo o número de pessoas do gênero masculino e fãs de beisebol pelo número total de fãs de beisebol. Isso representa a probabilidade de ser do gênero masculino, dado que a pessoa é fã de beisebol.

2. *Probabilidade de ser fã de basquete, dado que é do gênero feminino:*

```
prob = dist_conj.iloc[0, 0] / dist_conj.iloc[0, 4]
```

- `dist_conj.iloc[0, 0]`: Isso obtém a contagem de pessoas que são do gênero feminino e fãs de basquete na tabela de distribuição conjunta.
- `dist_conj.iloc[0, 4]`: Isso obtém a contagem total de pessoas que são do gênero feminino na tabela de distribuição conjunta.

A probabilidade condicional P(Basquete|Feminino) é calculada dividindo o número de pessoas fãs de basquete que são do gênero feminino pelo número total de pessoas do gênero feminino. Isso representa a probabilidade de ser fã de basquete, dado que a pessoa é do gênero feminino.

Distribuição conjunta normalizada pelas colunas:

```
''' Tabela de distribuição conjunta normalizada pelas colunas: Em nosso exemplo,
exibe todas as probabilidades condicionais do tipo P(Gênero|Esporte)'''
dist_conj_colunas = pd.crosstab(index=esportes['gênero'],
                                columns=esportes['esporte'],
                                margins=True, normalize='columns')
# Inspeção da tabela
display(dist_conj_colunas)
```

Neste trecho de código, você está construindo uma tabela de distribuição conjunta normalizada pelas colunas, que representa todas as probabilidades condicionais do tipo P(Gênero|Esporte). Vou explicar cada parte:

1. *Construção da tabela normalizada pelas colunas:*

```
dist_conj_colunas = pd.crosstab(index=esportes['gênero'],
                                columns=esportes['esporte'],
                                margins=True, normalize='columns')
```

- `pd.crosstab`: Isso cria uma tabela de contingência (distribuição conjunta) com base nas colunas do DataFrame `esportes`.

- `index=esportes['gênero']`: Define a variável do gênero como o índice da tabela.

- `columns=esportes['esporte']`: Define a variável do esporte como as colunas da tabela.

- `margins=True`: Isso adiciona uma linha e uma coluna marginais (total) à tabela.

- `normalize='columns'`: Isso normaliza a tabela pelas colunas, ou seja, calcula as probabilidades condicionais P(Gênero|Esporte).

2. *Exibição da tabela:*

```
display(dist_conj_colunas)
```

Isso exibe a tabela de distribuição conjunta normalizada pelas colunas, mostrando as probabilidades condicionais de gênero dado o esporte. Cada valor na tabela representa a probabilidade de uma determinada combinação de gênero e esporte, dividida pela probabilidade total do esporte correspondente. É uma maneira de entender como os gêneros estão distribuídos entre os diferentes esportes.

Distribuição conjunta normalizada pelas linhas (índice):

```
''' Tabela de distribuição conjunta normalizada pelas linhas: Em nosso exemplo,
exibe todas as probabilidades condicionais do tipo P(Esporte|Gênero)'''
dist_conj_linhas = pd.crosstab(index=esportes['gênero'],
                               columns=esportes['esporte'],
                               margins=True, normalize='index')
# Inspeção da tabela
display(dist_conj_linhas)
```

Neste trecho de código, você está construindo uma tabela de distribuição conjunta normalizada pelas linhas, que representa todas as probabilidades condicionais do tipo P(Esporte|Gênero). Vou explicar cada parte:

1. *Construção da tabela normalizada pelas linhas:*

```
dist_conj_linhas = pd.crosstab(index=esportes['gênero'],
                               columns=esportes['esporte'],
                               margins=True, normalize='index')
```

- `pd.crosstab`: Isso cria uma tabela de contingência (distribuição conjunta) com base nas colunas do DataFrame `esportes`.

- `index=esportes['gênero']`: Define a variável do gênero como o índice da tabela.

- `columns=esportes['esporte']`: Define a variável do esporte como as colunas da tabela.

- `margins=True`: Isso adiciona uma linha e uma coluna marginais (total) à tabela.

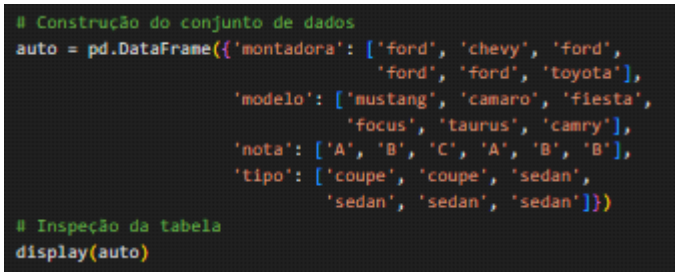
- `normalize='index'`: Isso normaliza a tabela pelas linhas, ou seja, calcula as probabilidades condicionais P(Esporte|Gênero).

2. *Exibição da tabela:*

```
display(dist_conj_linhas)
```

Isso exibe a tabela de distribuição conjunta normalizada pelas linhas, mostrando as probabilidades condicionais de esporte dado o gênero. Cada valor na tabela representa a probabilidade de um determinado esporte, dado o gênero, dividida pela probabilidade total do gênero correspondente. É uma maneira de entender como os esportes estão distribuídos entre os diferentes gêneros.

Probabilidades condicionais - automóveis:



Neste trecho de código, você está construindo um DataFrame chamado "auto" que contém informações sobre carros. Vou explicar cada parte:

1. *Construção do DataFrame "auto":*

```
auto = pd.DataFrame({'montadora': ['ford', 'chevy', 'ford',
                                   'ford', 'ford', 'toyota'],
                     'modelo': ['mustang', 'camaro', 'fiesta',
                                'focus', 'taurus', 'camry'],
                     'nota': ['A', 'B', 'C', 'A', 'B', 'B'],
                     'tipo': ['coupe', 'coupe', 'sedan',
                              'sedan', 'sedan', 'sedan']})
```

- `pd.DataFrame()`: Isso cria um DataFrame vazio.

- Os quatro dicionários dentro do `pd.DataFrame()` representam as colunas do DataFrame:

  - `'montadora'`: Representa a montadora do carro e contém uma lista de strings com nomes de montadoras.

  - `'modelo'`: Representa o modelo do carro e contém uma lista de strings com nomes de modelos.

- `'nota'`: Representa a nota atribuída ao carro e contém uma lista de strings com notas (A, B, C).

- `'tipo'`: Representa o tipo de carro (por exemplo, coupe, sedan) e contém uma lista de strings com os tipos.

2. *Exibição da tabela:*

```
display(auto)
```

Isso exibe o DataFrame "auto" na saída, permitindo que você veja os dados que foram inseridos nas colunas. O DataFrame contém informações sobre carros, como montadora, modelo, nota e tipo.

```
# Construção da tabela de distribuição conjunta
tabela = pd.crosstab(auto.tipo, auto.nota, margins=True)
# Inspeção da tabela
display(tabela)
# Cálculo das probabilidades
prob1 = tabela.iloc[1, 0] / tabela.iloc[1, 3]
print(f'P(A|sedan) = {prob1*100:.2f}%')
prob2 = tabela.iloc[0, 1] / tabela.iloc[2, 1]
print(f'P(coupe|B) = {prob2*100:.2f}%')
```

Neste trecho de código, você está realizando várias operações para analisar a distribuição conjunta de dados em um DataFrame chamado "auto." Vou explicar cada parte:

1. *Construção da tabela de distribuição conjunta:*

```
tabela = pd.crosstab(auto.tipo, auto.nota, margins=True)
```

- `pd.crosstab(auto.tipo, auto.nota, margins=True)`: Isso cria uma tabela de distribuição conjunta (também chamada de tabela de contingência) entre as colunas "tipo" e "nota" do DataFrame "auto." A tabela resultante mostra a frequência de ocorrência de cada combinação de valores entre essas duas colunas. O argumento `margins=True` adiciona as margens à tabela, mostrando totais nas linhas e colunas.

2. *Inspeção da tabela:*

```
display(tabela)
```

Isso exibe a tabela de distribuição conjunta "tabela" na saída para que você possa visualizar os valores.

3. *Cálculo de probabilidades condicionais:*

```
prob1 = tabela.iloc[1, 0] / tabela.iloc[1, 3]
print(f'P(A|sedan) = {prob1*100:.2f}%')
```

- `tabela.iloc[1, 0]` e `tabela.iloc[1, 3]`: Essas linhas calculam a frequência da combinação "sedan" e "A" (nota A) e o total de sedans na tabela de distribuição conjunta.

- `prob1 = tabela.iloc[1, 0] / tabela.iloc[1, 3]`: Isso calcula a probabilidade condicional de um carro ser "A" (nota A) dado que é um "sedan." O resultado é exibido com duas casas decimais.

```
prob2 = tabela.iloc[0, 1] / tabela.iloc[2, 1]
print(f'P(coupe|B) = {prob2*100:.2f}%')
```

- `tabela.iloc[0, 1]` e `tabela.iloc[2, 1]`: Essas linhas calculam a frequência da combinação "coupe" e "B" (nota B) e o total de carros com nota B na tabela de distribuição conjunta.

- `prob2 = tabela.iloc[0, 1] / tabela.iloc[2, 1]`: Isso calcula a probabilidade condicional de um carro ser "coupe" dado que possui nota B. O resultado é exibido com duas casas decimais.

Em resumo, você está calculando probabilidades condicionais com base na tabela de distribuição conjunta entre os tipos de carros e as notas atribuídas.

```
# Probabilidades condicionais: P(nota|tipo)
print('** Probabilidades condicionais: P(nota|tipo) **')
tabela_linhas = pd.crosstab(auto.tipo, auto.nota,
                            margins=True, normalize="index")
display(tabela_linhas)
# Probabilidades condicionais: P(tipo|nota)
print('** Probabilidades condicionais: P(tipo|nota) **')
tabela_colunas = pd.crosstab(auto.tipo, auto.nota,
                             margins=True, normalize="columns")
display(tabela_colunas)
```

Nesse trecho de código, você está calculando probabilidades condicionais considerando a tabela de distribuição conjunta entre as colunas "tipo" e "nota" do DataFrame "auto." Aqui estão as explicações para cada parte do código:

Probabilidades condicionais: P(nota|tipo)

```
tabela_linhas = pd.crosstab(auto.tipo, auto.nota, margins=True, normalize="index")
display(tabela_linhas)
```

- `tabela_linhas`: Isso cria uma nova tabela de distribuição, normalizada pelas linhas, ou seja, você está calculando as probabilidades condicionais P(nota|tipo), ou seja, a probabilidade de uma nota dada um certo tipo de carro.
- `pd.crosstab(auto.tipo, auto.nota, margins=True, normalize="index")`: Aqui, você está utilizando o método `pd.crosstab` para criar a tabela de distribuição condicional. O argumento `normalize="index"` garante que as probabilidades sejam normalizadas pelas linhas, ou seja, a soma das probabilidades em cada linha será igual a 1. O argumento `margins=True` adiciona margens à tabela para mostrar as somas.

Probabilidades condicionais: P(tipo|nota)

```
tabela_colunas = pd.crosstab(auto.tipo, auto.nota, margins=True, normalize="columns")
display(tabela_colunas)
```

- `tabela_colunas`: Aqui, você está criando outra tabela de distribuição condicional, mas desta vez normalizada pelas colunas, ou seja, você está calculando as probabilidades condicionais P(tipo|nota), ou seja, a probabilidade de um tipo de carro dado um certo nota.
- `pd.crosstab(auto.tipo, auto.nota, margins=True, normalize="columns")`: Mais uma vez, você utiliza o método `pd.crosstab` para criar a tabela de distribuição condicional. O argumento `normalize="columns"` garante que as probabilidades sejam normalizadas pelas colunas, ou seja, a soma das probabilidades em cada coluna será igual a 1. O argumento `margins=True` adiciona margens à tabela para mostrar as somas.

Essas tabelas de distribuição condicional são úteis para analisar como as notas e os tipos de carros estão relacionados e para calcular as probabilidades condicionais correspondentes.