Conjunto de dados - Notas obtidas em três provas

- est_provas = pd.DataFrame(tabela.mean(axis=0, numeric_only=True), columns=['Média'])
  est_provas['Mediana'] = tabela.median(axis=0, numeric_only=True)
  est_provas['DesvPad'] = tabela.std(axis=0, ddof=0, numeric_only=True)
  amps = [tabela['P'+str(i)].max() - tabela['P'+str(i)].min() for i in range(1,4)]
  est_provas['Amplitudes'] = amps
  display(est_provas)

- Este trecho calcula várias estatísticas descritivas para cada uma das três provas (P1, P2 e P3) e as armazena no DataFrame **est_provas**.
- **tabela.mean(axis=0, numeric_only=True)** calcula a média de cada coluna numérica (as colunas de notas) e cria uma coluna chamada 'Média' em **est_provas**.
- **tabela.median(axis=0, numeric_only=True)** calcula a mediana de cada coluna numérica e adiciona uma coluna 'Mediana' em **est_provas**.
- **tabela.std(axis=0, ddof=0, numeric_only=True)** calcula o desvio padrão de cada coluna numérica com **ddof=0** (população) e cria uma coluna 'DesvPad' em **est_provas**.
- O loop **for i in range(1,4)** calcula a amplitude (diferença entre o valor máximo e mínimo) para cada prova e armazena esses valores em uma lista chamada **amps**. Essa lista é usada para criar a coluna 'Amplitudes' em **est_provas**.
- **display(est_provas)** exibe o DataFrame **est_provas** que contém todas essas estatísticas descritivas.

- **Identificação da Prova mais Homogênea**:
- homogenea = est_provas.loc[est_provas['DesvPad'] == est_provas['DesvPad'].min()]
  print('*** Prova mais homogênea ***')
  display(homogenea)

- Este trecho identifica a prova mais homogênea com base no menor desvio padrão (**DesvPad**) no DataFrame **est_provas**. O resultado é armazenado em **homogenea**.
- Em seguida, ele imprime "*** Prova mais homogênea ***" e exibe o DataFrame **homogenea**, que contém a prova mais homogênea e suas estatísticas.

- **Uso do Método agg**:
- print('Uso do método agg')
  tabela.agg(['mean', 'std', 'median'])

- Este trecho usa o método **agg** no DataFrame **tabela** para calcular as mesmas estatísticas descritivas (média, desvio padrão e mediana) para todas as colunas numéricas do DataFrame. O resultado é impresso na tela.

```
# Médias de desvios padrão por aluno
''' É preciso usar axis=1 para o cálculo nas linhas do DataFrame.
CUIDADO: o retorno é um objeto Series, logo é preciso usasar list() para
armazenar em uma coluna de um DataFrame.'''
est_alunos = pd.DataFrame(tabela.mean(axis=1, numeric_only=True), columns=
['Média'])
est_alunos.index = list(tabela['Aluno'])
est_alunos['DesvPad'] = list(tabela.std(axis=1, ddof=0, numeric_only=True))
display(est_alunos)

# Encontrando o melhor aluno
print('*** Melhor Aluno ***')
melhor = est_alunos.loc[est_alunos['Média'] == est_alunos['Média'].max()]
display(melhor)

# Encontrando o aluno mais consistente
print('*** Aluno mais consistente ***')
consistente = est_alunos.loc[est_alunos['DesvPad'] == est_alunos['DesvPad'].
min()]
display(consistente)
```

- est_alunos = pd.DataFrame(tabela.mean(axis=1, numeric_only=True), columns=['Média'])
  est_alunos.index = list(tabela['Aluno'])
  est_alunos['DesvPad'] = list(tabela.std(axis=1, ddof=0, numeric_only=True))
  display(est_alunos)

- Este trecho calcula estatísticas descritivas para cada aluno com base em suas notas nas três provas.
- **tabela.mean(axis=1, numeric_only=True)** calcula a média das notas de cada aluno (ao longo das colunas) e cria uma coluna chamada 'Média' no DataFrame **est_alunos**.
- **est_alunos.index = list(tabela['Aluno'])** define os nomes dos alunos como índices no DataFrame **est_alunos**, usando a coluna 'Aluno' da tabela original.
- **list(tabela.std(axis=1, ddof=0, numeric_only=True))** calcula o desvio padrão das notas de cada aluno (ao longo das colunas) com **ddof=0** (população) e cria uma coluna 'DesvPad' no DataFrame **est_alunos**.
- **display(est_alunos)** exibe o DataFrame **est_alunos** que contém essas estatísticas descritivas para cada aluno.

- **Identificação do Melhor Aluno**:
- print('*** Melhor Aluno ***')
  melhor = est_alunos.loc[est_alunos['Média'] == est_alunos['Média'].max()]
  display(melhor)

- Este trecho identifica o melhor aluno com base na maior média (**'Média'**) no DataFrame **est_alunos**.
- Ele imprime "*** Melhor Aluno ***" e exibe o DataFrame **melhor**, que contém o aluno com a maior média.

- **Identificação do Aluno mais Consistente**:
- print('*** Aluno mais consistente ***')
  consistente = est_alunos.loc[est_alunos['DesvPad'] ==
  est_alunos['DesvPad'].min()]
  display(consistente)

- Este trecho identifica o aluno mais consistente com base no menor desvio padrão (**'DesvPad'**) no DataFrame **est_alunos**.
- Ele imprime "*** Aluno mais consistente ***" e exibe o DataFrame **consistente**, que contém o aluno com o menor desvio padrão.

Média e desvio padrão de dados agrupados

```
# Colunas iniciais da distribuição de frequência
dados = pd.DataFrame({'Classe': ['0-4', '5-9', '10-14', '15-19',
                                 '20-24', '25-29', '30 ou +'],
                      'Freq (f)': [5, 12, 24, 17, 16, 11, 5],
                      'Pt. médio (x)':[2, 7, 12, 17, 22, 27, 32]})
# Coluna valor * frequência
dados['x*f'] = dados['Freq (f)']*dados['Pt. médio (x)']
# Cálculo da média
media_dados = dados['Pt. médio (x)'].mean()
# Coluna valor - média
dados['x-xbarra'] = dados['Pt. médio (x)'] - media_dados
# Coluna (valor - média)^2
dados['(x-xbarra)^2'] = dados['x-xbarra']**2
# Coluna (valor - média)^2 * frequência
dados['(x-xbarra)^2*f'] = dados['(x-xbarra)^2']*dados['Freq (f)']
display(dados)
# Cálculo da média amostral
obs = dados['Freq (f)'].sum()
media_am = dados['x*f'].sum()/obs
print(f'Média amostral: {media_am:.1f} horas')
# Cálculo do desvio padrão amostral
desv_pam_am = (dados['(x-xbarra)^2*f'].sum()/(obs-1))**(1/2)
print(f'Desvio padrão amostral: {desv_pam_am:.1f} horas')
```

- **Cálculo da Coluna x*f**:
- dados['x*f'] = dados['Freq (f)']*dados['Pt. médio (x)']

- Aqui, o código multiplica a frequência de cada classe pelo ponto médio da classe e armazena o resultado na coluna **'x*f'**.

- **Cálculo da Média Amostral**:
- media_dados = dados['Pt. médio (x)'].mean()
  obs = dados['Freq (f)'].sum()
  media_am = dados['x*f'].sum()/obs

- O código calcula a média dos pontos médios das classes e a armazena em **media_dados**.
- Ele também calcula o número total de observações (soma das frequências) e, em seguida, calcula a média amostral dividindo a soma de **'x*f'** pelo número total de observações. O resultado é armazenado em **media_am**.

- **Cálculo do Desvio Padrão Amostral**:
- dados['x-xbarra'] = dados['Pt. médio (x)'] - media_dados
  dados['(x-xbarra)^2'] = dados['x-xbarra']**2
  dados['(x-xbarra)^2*f'] = dados['(x-xbarra)^2']*dados['Freq (f)']
  desv_pam_am = (dados['(x-xbarra)^2*f'].sum()/(obs-1))**(1/2)

- O código calcula a diferença entre cada ponto médio da classe e a média (**'x-xbarra'**) e armazena essa diferença em uma nova coluna.
- Em seguida, ele calcula o quadrado dessa diferença (**'(x-xbarra)^2'**) e armazena o resultado em outra coluna.

- Multiplica o quadrado da diferença pelo número de observações (**'(x-xbarra)^2*f'**) e armazena o resultado em uma nova coluna.
- Finalmente, calcula o desvio padrão amostral usando a fórmula padrão e o resultado é armazenado em **desv_pam_am**.

Comparando dois conjuntos de dados

```
# Conjunto de dados
salarios = pd.DataFrame({'Dallas': [41.6, 50.0, 49.5, 38.7, 39.9,
                                    45.8, 44.7, 47.8, 40.5, 44.3],
                         'New York': [45.6, 41.5, 57.6, 55.1, 59.3,
                                      59.0, 50.6, 47.2, 42.3, 51.0]})
display(salarios)

# Cálculo das médias, desvios padrão e coeficiente de variação
est_sal =pd.DataFrame(salarios.mean(axis=0, numeric_only=True), columns=
['Média'])
est_sal['DesvPad'] = salarios.std(axis=0, ddof=0, numeric_only=True)
est_sal['CV'] = est_sal['DesvPad']/est_sal['Média']*100
display(est_sal)
```

- **Cálculo de Estatísticas Descritivas**:
- est_sal =pd.DataFrame(salarios.mean(axis=0, numeric_only=True), columns=['Média'])
  est_sal['DesvPad'] = salarios.std(axis=0, ddof=0, numeric_only=True)
  est_sal['CV'] = est_sal['DesvPad']/est_sal['Média']*100

- O código calcula a média das colunas **'Dallas'** e **'New York'** usando **salarios.mean(axis=0, numeric_only=True)**. O resultado é armazenado em uma coluna chamada 'Média' no DataFrame **est_sal**.
- Em seguida, calcula o desvio padrão das colunas **'Dallas'** e **'New York'** com **ddof=0** (população) usando **salarios.std(axis=0, ddof=0, numeric_only=True)** e armazena o resultado em uma coluna chamada 'DesvPad' em **est_sal**.
- Calcula o coeficiente de variação (CV) para cada cidade dividindo o desvio padrão pela média e multiplicando por 100. O resultado é armazenado em uma coluna chamada 'CV' em **est_sal**.