Pandas

Ferramentas para Data Science

Carolina Ribeiro Xavier
CCOMP - UFSJ

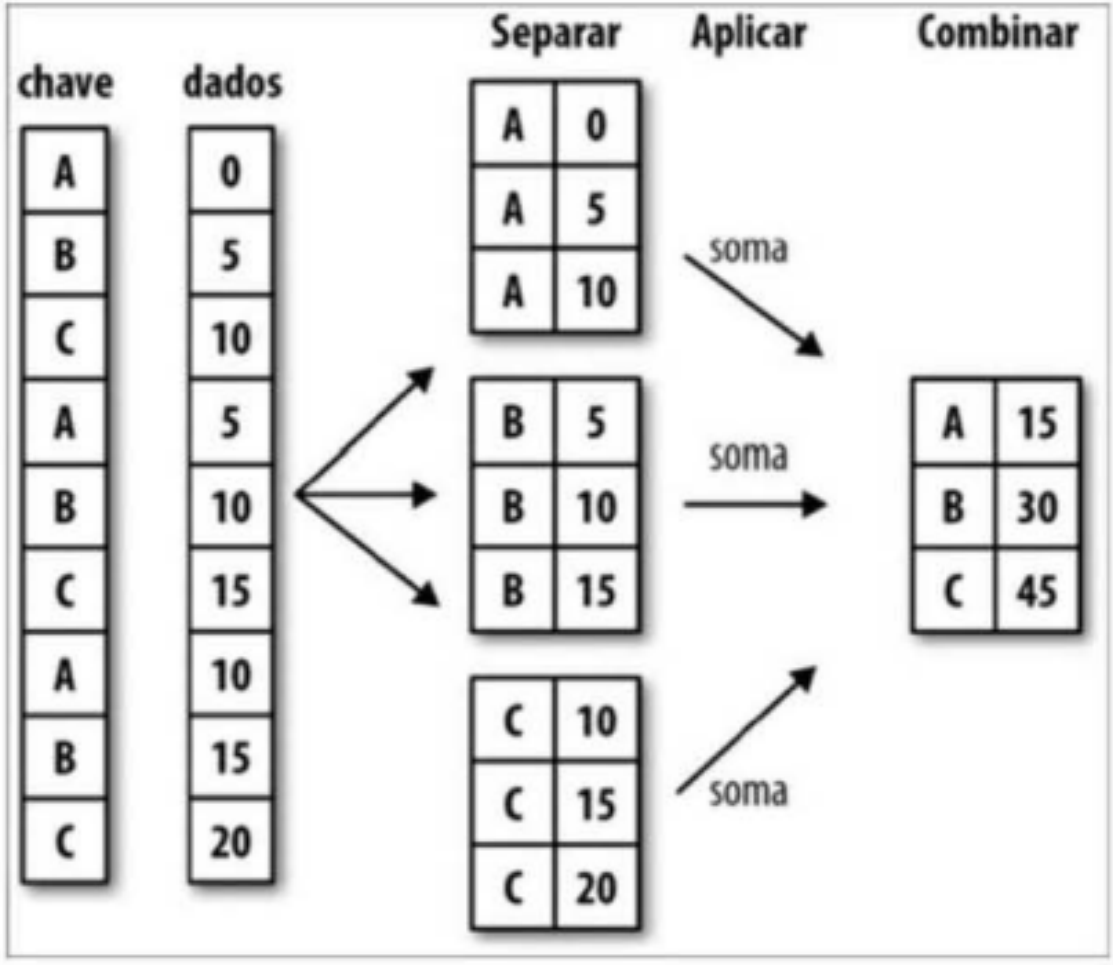
Motivação para Agrupamento de Dados

- Análise de dados frequentemente requer resumos por categorias.
- O agrupamento facilita operações específicas para cada categoria.
- Pandas permite o agrupamento eficiente e manipulação desses dados.

Estrutura do Método `groupby`

- O método `groupby` facilita operações de agregação e transformação.
- Principais etapas:
  1. Dividir os dados em grupos.
  2. Aplicar operações em cada grupo.
  3. Combinar resultados em um novo objeto.

groupby




```
DataFrame.groupby(by=None, axis=<no_default>, level=None,
as_index=True, sort=True, group_keys=True, observed=<no_default>
dropna=True)
```

Sintaxe Básica

```
import pandas as pd
df = pd.DataFrame({
    'Equipe': ['A', 'B', 'A', 'B'],
    'Pontos': [10, 20, 15, 25]
})

grupo = df.groupby('Equipe')
grupo.mean()
```

- Agrupamento por coluna "Equipe" e cálculo da média.

```
Média dos Pontos por Equipe:
        Pontos
Equipe
A         12.5
B         22.5
```

Iterando sobre grupos

```python
for name, group in df.groupby('Equipe'):
    print(name)
    print(group)
```

```
A
  Equipe  Pontos
0      A      10
2      A      15
B
  Equipe  Pontos
1      B      20
3      B      25
```

Selecionando um grupo

```
grouped = df.groupby("Equipe")
print(grouped.get_group("A"))
```

```
  Equipe  Pontos
0      A      10
2      A      15
```

Operações Básicas com GroupBy

- `mean()`: Média dos valores.
- `sum()`: Soma dos valores.
- `count()`: Contagem de elementos em cada grupo.
- `min()`/`max()`: Valores mínimo e máximo.

Exemplo com Operações Básicas

```
grupo.sum()
```

- Soma os valores de cada grupo.

```
Soma dos Pontos por Equipe:
         Pontos
Equipe
A            25
B            45
```

Exemplo com Operações Básicas

```
print("\nContagem de Entradas por Equipe:")
print(grupo.count())
print("\nValor Mínimo e Máximo de Pontos por Equipe:")
print("Mínimo:\n", grupo.min())
print("Máximo:\n", grupo.max())
```

```
Contagem de Entradas por Equipe:
        Pontos
Equipe
A            2
B            2
Valor Mínimo e Máximo de Pontos por Equipe:
Mínimo:
         Pontos
Equipe
A           10
B           20
Máximo:
         Pontos
Equipe
A           15
B           25
```

Agrupamento por Múltiplas Colunas

- Podemos agrupar por mais de uma coluna.
- Cria uma estrutura hierárquica.

Exemplo de Agrupamento Múltiplo

```
df = pd.DataFrame({
    'Equipe': ['A', 'A', 'B', 'B'],
    'Ano': [2020, 2021, 2020, 2021],
    'Pontos': [10, 15, 20, 25]
})

grupo = df.groupby(['Equipe', 'Ano']).mean()
print(grupo)

Média de Pontos por Equipe e Ano:
              Pontos
Equipe Ano
A      2020     10.0
       2021     15.0
B      2020     20.0
       2021     25.0
```

Aplicação de Funções Customizadas

- `apply()` permite aplicar funções específicas em cada grupo.

Exemplo com Função Customizada

```
def range_func(x):
    return x.max() - x.min()

grupo = df.groupby('Equipe')['Pontos'].apply(range_func)
print(grupo)

Diferença entre o Máximo e Mínimo de Pontos por Equipe:
Equipe
A    5
B    5
Name: Pontos, dtype: int64
```

Agregação com Várias Funções

- `agg()` permite aplicar múltiplas funções simultaneamente.

Exemplo com agg

```
grupo = df.groupby('Equipe').agg({
    'Pontos': ['sum', 'mean'],
    'Assistências': 'max'
})
print(grupo)
```

```
Agregação com Várias Funções:
       Pontos        Assistências
          sum  mean           max
Equipe
A          30  15.0             5
B          40  20.0             6
```

Filtragem de Grupos com GroupBy

- `filter()` seleciona grupos com base em uma condição.

Exemplo de Filtragem

```
grupo = df.groupby('Equipe').filter(lambda x: x['Pontos'].mean() > 12)
print(grupo)

Filtragem de Grupos com Média de Pontos > 12:
  Equipe  Pontos
0      A      10
1      B      20
2      A      15
3      B      25
```

Transformação com `transform`

- `transform()` aplica uma função a cada grupo e devolve um DataFrame.

Exemplo de Transformação

```
df['Pontos_Ajustados'] =
df.groupby('Equipe')['Pontos'].transform(lambda x: x - x.mean())
print(df)

DataFrame com Pontos Ajustados por Equipe:
  Equipe  Pontos  Pontos_Ajustados
0      A      10              -2.5
1      B      20              -2.5
2      A      15               2.5
3      B      25               2.5
```

Operações Avançadas de GroupBy

- Utilizando várias colunas e operações complexas.

Usando Funções Estatísticas

```
grupo = df.groupby('Equipe')['Pontos'].describe()
print(grupo)

Estatísticas Descritivas dos Pontos por Equipe:
         count  mean       std   min    25%   50%    75%   max
Equipe
A          2.0  12.5  3.535534  10.0  11.25  12.5  13.75  15.0
B          2.0  22.5  3.535534  20.0  21.25  22.5  23.75  25.0
```

- `describe()` fornece estatísticas detalhadas para cada grupo.

Seminários da disciplina - proposta de datas

Antes da prática...

- **Ferramentas: 12 e 14 de Novembro:**
  Escolher uma ferramenta para python que tenha utilidade para análise e visualização de dados, fazer uma apresentação da ferramenta de até 20 minutos. (enviar previamente em planilha e aguardar o meu ok)
- **Artigo: 09 e 11 de Dezembro:**
  Apresentação de um artigo que faça análise grande quantidade de dados com uso de técnicas de aprendizado de máquina ou que inova na visualização de dados. (enviar previamente em planilha e aguardar o meu ok)

Exercícios Práticos

Inicia hoje e tem até segunda para entregar, estarei a disposição no horário da aula para ajudar os que não conseguirem até lá. Quem entregar está liberado da frequência de segunda-feira.
Aplicar os conceitos do método groupby e funções adjacentes, como agg, filter, apply, entre outras, para explorar fatores que influenciam o desempenho dos alunos.
Instruções:
Carregar e Preparar o Dataset:
Carregue o arquivo StudentPerformanceFactors.csv. Inspecione as primeiras linhas e obtenha informações gerais sobre o dataset, como tipos de dados e valores ausentes.
Em seguida calcule a distribuição de Notas por Categoria, use groupby para agrupar os dados por uma variável categórica de seu interesse e calcule a média e o desvio padrão das notas dos alunos.
Compare as Categorias criando uma comparação entre as médias das notas, ordenando o resultado para identificar quais grupos possuem melhor desempenho médio.

Exercícios Práticos

Faça uma correlação entre Fatores e Desempenho, para isso, agrupe os dados por uma variável categórica, como "Tipo de Curso" ou "Horário de Estudo", e analise como essas variáveis influenciam as notas dos alunos. Utilize o método agg para obter múltiplas estatísticas (média, mediana e contagem de alunos). Visualize esses resultados com gráficos, como box plots ou gráficos de barras, para evidenciar as diferenças entre os grupos.
Crie um filtro para encontrar grupos de alunos que atendam a determinadas condições (ex.: alunos com mais de uma certa média e em uma categoria específica). Utilize o filter para identificar grupos que atendem a uma condição personalizada (ex.: cursos com mais de 20 alunos que possuem média de desempenho superior a uma determinada nota).

Exercícios Práticos

Utilize o método apply para calcular uma métrica personalizada sobre os grupos. Por exemplo, crie uma função que classifique os alunos em diferentes níveis de desempenho (ex.: "Alto", "Médio", "Baixo") com base na média de notas de cada grupo e aplique-a aos dados.
Gere um relatório consolidado com gráficos e tabelas que resumem as principais descobertas da análise. Inclua, pelo menos essas:
Comparações de médias entre grupos
Grupos que mais se destacam em termos de desempenho
Análise dos principais fatores que impactam o desempenho (ex.: Tipo de Curso, Gênero, Horário de Estudo)
Use visualizações de dados, como gráficos de barras ou boxplots, para dar suporte às conclusões. (Explore a biblioteca Seaborn)