Programação e Análise de Dados com Python

Programa de Pós-Graduação em Economia - PPGE

Python - Conjuntos e aplicações

Prof. Hilton Ramalho

Prof. Aléssio Almeida

Objetivo

Apresentar noções gerais de operações com conjuntos no `Python`.

Conteúdo

Conjuntos

- Esses objetos é **não admitem repetições de elementos**, diferente de listas e tuplas.
- Conjuntos não mantêm a ordem de seus elementos.
- Conjuntos não aceitam **elementos mutáveis como listas**.
- Conjuntos **são mutáveis**, você pode adicionar e remover elementos dele.
- Conjuntos também podem ser usados para efetuar operações matemáticas como união, interseção, etc.
- `help(set)`

- Para criar um conjunto devemos usar a função `set()` passando uma lista de elementos. Também é possível criar um conjunto usando `{ }` e passando os elementos. No entanto, esse último método não é recomendável uma vez que pode também ser usado para criação de outro tipo de objeto denotado de **dicionário**.

Métodos aplicáveis a objetos conjuntos (set)

Método	Descrição
add()	Adiciona um elemento ao conjunto.
clear()	Remove todos os elementos de um conjunto.
copy()	Retorna uma cópia do conjunto.
difference()	Returna um novo conjunto que é a diferença entre dois ou mais conjuntos.

Método	Descrição
difference_update()	Remove todos os elementos de um outro conjunto para esse conjunto.
discard()	Removes an element from the set if it is a member. (Do nothing if the element is not in set)
intersection()	Returns the intersection of two sets as a new set
intersection_update()	Updates the set with the intersection of itself and another
isdisjoint()	Returns True if two sets have a null intersection
issubset()	Returns True if another set contains this set
issuperset()	Returns True if this set contains another set
pop()	Removes and returns an arbitrary set element. Raises KeyError if the set is empty
remove()	Removes an element from the set. If the element is not a member, raises a KeyError
symmetric_difference()	Returns the symmetric difference of two sets as a new set
symmetric_difference_update()	Updates a set with the symmetric difference of itself and another
union()	Returns the union of sets in a new set

In [ ]:
```
help(set)
```

In [ ]:
```
# Criando um set com { .... }
# Usar com cautela para não gerar conflito com dicionários
s = { 1, 2, 3}
print(s, type(s))
```

```
{1, 2, 3} <class 'set'>
```

In [ ]:
```
# Criando um set usando a função set() e passando uma lista de elementos. F
s = set( [1,2,3,3] )
type(s)
print(f"O conjunto {s} é da classe {type(s)}")
```

```
O conjunto {1, 2, 3} é da classe <class 'set'>
```

In [ ]:
```
# Criando um set com elementos repetidos ?
s = set( [1, 1, 1, 3, 3, 3, 1, 2] )
# Serão mantidas as ocorrencias únicas desses elementos
print(s)
```

```
{1, 2, 3}
```

In [ ]:
```
# Criando um conjunto vazio
s = set()
print(s,type(s))
```

```
set() <class 'set'>
```

In [ ]:
```
# Criando um conjunto com elementos texto, número e tuplas
s = set( [1, 'a', (1, 3), 2] )
print(f"{s} é um conjunto {type(s)}")
```

```
{1, 2, (1, 3), 'a'} é um conjunto <class 'set'>
```

In [ ]:
```
# Criando um conjunto com elementos texto, número e tuplas
s = set( [1, 'a', (1, 3), (1,3), 2, 'a'] )
print(f"{s} é um conjunto {type(s)}")
```

```
{1, 2, (1, 3), 'a'} é um conjunto <class 'set'>
```

In [ ]:
```
# Tente criar um set com elementos lista
s = set([1, 3, [2,3]])
```

In [ ]:
```
# Conjunto não suporta indexação de elementos
s = set([1,2,3])
s[2]
```

Posição de elementos nos conjuntos

- Posição não importa.
- Não é possível usar a indexação para frente ou inversa para acessar elementos de um conjunto.

In [ ]:
```
w = {1, 3, 5}
w[0]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-7-04ea69447b14> in <module>()
      1 w = {1, 3, 5}
----> 2 w[0]

TypeError: 'set' object is not subscriptable
```

Método .add() - Adicionando elementos a um conjunto

- Usamos o método `.add()`.
- Veja `help(set)`.

In [ ]:
```
# Adicionando elementos ao conjunto vazio
s = set()
# Novos elementos
s.add(2)
s.add(3)
print(s)
```

```
{2, 3}
```

In [ ]:
```
# Adicionando elementos ao conjunto vazio
s = set()
print(s)
# Novos elementos
s.add(('a','b'))
s.add(2)
print(s)
```

```
set()
{2, ('a', 'b')}
```

Método .update() - Adicionando múltiplos elementos a um conjunto

- Usamos o método `.update()` passando os elementos em listas ou sets.
- Veja `help(set)`.

In [ ]:
```
# Novo conjunto
s = set()
# Adicionar um elemento
s.add(2)
# Adicionar múltiplos elementos - devem ser passados em uma lista.
s.update([4,5,8])
print(s)
```

```
{8, 2, 4, 5}
```

In [ ]:
```
# Novo conjunto
s = set()
# Adicionar um elemento
s.add(0)
# Adicionar múltiplos elementos - devem ser passados em uma lista ou outro
s.update([3,5], {1,3,5})
print(s)
```

Método .remove()

- Remove determinado elemento um conjunto.

Usamos o método `.remove()` passando o elemento. Veja help(set).

In [ ]:
```
# Exemplo
s = set([1,2,2,4,4,5,6,7])
print(s)
```

```
{1, 2, 4, 5, 6, 7}
```

In [ ]:
```
# Remover o elemento 1
s.remove(1)
print(s)
```

```
{2, 4, 5, 6, 7}
```

Principais operações com conjuntos

- União de conjuntos. Usamos o método `.union()`

In [ ]:
```
# Exemplo
a = set([1, 2, 5])
b = set([3, 2, 4, 8])

# Criamos um novo conjunto a partir da união dos conjuntos a e b
c = a.union(b)

print(f"O conjunto {c} é a união de {a} e {b}")
```

```
O conjunto {1, 2, 3, 4, 5, 8} é a união de {1, 2, 5} e {8, 2, 3, 4}
```

Principais operações com conjuntos

- Intersecção de conjuntos. Usamos o método `.intersection()`

In [ ]:
```
# Exemplo
a = set([1, 2, 5])
b = set([3, 2, 4, 8])

# Criamos um novo conjunto a partir da intersecção dos conjuntos a e b
c = a.intersection(b)

print(f"O conjunto {c} é a intersecção de {a} e {b}")
```

```
O conjunto {2} é a intersecção de {1, 2, 5} e {8, 2, 3, 4}
```

Principais operações com conjuntos

- Diferença de conjuntos. Usamos o método `.difference()`
- Dados os conjuntos A e B. A diferença A - B é conjunto de elementos de A que não pertencem a B.

In [ ]:
```
# Exemplo
w = set([1,2,3,5,9])
z = set([1,0,4,0])

print(f"Conjunto w={w} e conjunto z={z}")
```

```
Conjunto w={1, 2, 3, 5, 9} e conjunto z={0, 1, 4}
```

In [ ]:
```
# Criando o conjunto diferença
c = a.difference(b)
print(f"c={c} é o conjunto diferença entre {a} e {b}")
```

```
c={1, 5} é o conjunto diferença entre {1, 2, 5} e {8, 2, 3, 4}
```

Métodos de conjuntos para estruturas de decisão

- Alguns métodos de conjuntos podem ser usados para regras de decisão.
- Por exemplo, os métodos `isdisjoint`, `issubset` e `issuperset`.

In [ ]:
```
# Exemplos
w = set([1,3,3,2,9])
z = set([1,2])

print(f"Conjunto w={w} e conjunto z={z}")
```

```
Conjunto w={1, 2, 3, 9} e conjunto z={1, 2}
```

In [ ]:
```
# O conjunto W é disjunto de Z (intersecção é um conjunto vazio) ?
w.isdisjoint(z)
```

In [ ]:
```
# O conjunto W é subconjunto de Z (está contido) ?
w.issubset(z)
```

In [ ]:
```
# O conjunto Z é subconjunto de W (está contido) ?
w.issubset(z)
```

In [ ]:
```
# O conjunto W é contém Z (é super conjunto) ?
w.issuperset(z)
```

In [ ]:
```
# O conjunto vazio é subconjunto de W ?
set().issubset(w)
```

In [ ]:
```
# Conjunto {2,3}  é subconjunto de W ?
set([2,3]).issubset(w)
```

In [ ]:
```
# Usando como regra de decisão
q = set([2,3])
if q.issubset(w):
  print(f"O conjunto {q} está contido em {w}")
```

```
O conjunto {2, 3} está contido em {1, 2, 3, 9}
```

In [ ]:
```
# Usando como regra de decisão
q = set([1,2,3,4,7,9,0])
if q.issuperset(w):
  print(f"O conjunto {q} contém o conjunto {w}")
```

```
O conjunto {0, 1, 2, 3, 4, 7, 9} contém o conjunto {1, 2, 3, 9}
```

Operadores relacionais

- Usamos `in` ou `not in` para testar se determinado elemento pertence ou não a um conjunto.

In [ ]:

```
# Testando se determinado elemento pertence a um conjunto

s = set([1,4,6])

if 4 in s:
  print(f"O elemento 4 percente ao conjunto s={s}")
```

In [ ]:

```
# Testando se determinado elemento não pertence a um conjunto

s = set([1,4,6])

if 5 not in s:
  print(f"O elemento 5 não percente ao conjunto s={s}")
else:
  print("Sim")
```

```
O elemento 5 não percente ao conjunto s={1, 4, 6}
```