Big Data & Data Science

Vetores n-dimensionais

Arrays em Python

import array

- Declaração:
  - MeuVetor = array(tipo, [inicializações])
  - MeuVetor = array(‘c’, [‘a’, ‘b’, ‘c’])
  - MeuVetor = array(‘i’, [1, 2, 3])
  - MeuVetor = array(‘f’, [1.0, 2.0, 3.0])

  - >>> type(MeuVetor)
  - <class 'array.array'>

Inserção de elementos

```
from array import *
>>> MeuVetor = array('i', [1, 2, 3])
>>> MeuVetor
array('i', [1, 2, 3])

>>> MeuVetor.append(4)
>>> MeuVetor
array('i', [1, 2, 3, 4])
```

```
>>> MeuVetor.insert(0, 0)
>>> MeuVetor
array('i', [0, 1, 2, 3, 4])
```

Inserção de elementos

```
>>> lista = ["a", "b"]
>>> MeuVetor = array('i', [1, 2, 3])
```

```
>>> MeuVetor.fromlist(lista)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: an integer is required (got type str)
```

```
>>> lista = ["a", "b"]
>>> MeuVetor = array('i', [1, 2, 3])
```

```
>>> MeuVetor.append("a")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: an integer is required (got type str)
```

Remoção de elementos

```
from array import *
>>> MeuVetor = array('i', [0, 1, 2, 3])
```

```
>>> MeuVetor.remove(2)
>>> MeuVetor
array('i', [0, 1, 3])
```

```
>>> MeuVetor.pop()
>>> MeuVetor
array('i', [0, 1])
```

```
>>> MeuVetor.pop(0)
>>> MeuVetor
array(‘i’, [1])
```

Expansão do vetor

```
>>> lista = [7, 8, 9]
>>> MeuVetor = array('i', [1, 2, 3])


>>>MeuVetor.fromlist(lista)


>>> MeuVetor
array('i', [1, 2, 3, 7, 8, 9])
```

Expansão do vetor

```
>>> outroVetor = array(‘i’, [4, 5, 6])
>>> MeuVetor = array('i', [1, 2, 3])

>>>MeuVetor.extend(outroVetor)

>>> MeuVetor
array('i', [1, 2, 3, 4, 5, 6])
```

Outros métodos sobre vetor

```
>>> MeuVetor
array('i', [1, 2, 3, 2, 2])

>>> MeuVetor.reverse( )
>>> MeuVetor.count(2)
>>> MeuVetor.index(3)
>>> len(MeuVetor)
```

Soma dos elementos

```
>>> def soma(vetor):
...     total = 0
...     for e in vetor:
...         total += e
...     return total
...
>>> soma(MeuVetor)
```

OU

```
sum(MeuVetor)
```

NumPy array

Biblioteca “Numerical Python”

- Núcleo de funções para computação científica
- Estrutura de dados principal: array

NumPy array:

- Objeto de vetor multidimensional de alto desempenho
- Computação eficiente de vetores e matrizes

NumPy array

import numpy as np

- A classe vetor do numpy é chamada “ndarray”
- Diferente da classe “array” do Python
  - Mais funcionalidades
  - Lida com mais de uma dimensão
- Um vetor numpy consiste de:
  - Uma tabela de elementos do mesmo tipo (em geral numéricos)
  - Uma tupla de inteiros positivos que indexa tais elementos
  - Suas dimensões são chamadas de *eixos* (axis)

Criando um vetor NumPy

```
import numpy as np
>>> a = np.array([3., 4., 5.])
>>> a
Array([3., 4., 5.])

>>> type(a)
<class 'numpy.ndarray'>
```

Vetor NumPy: atributos

- Número de dimensões/eixos:

```
>>> b = np.array([[0, 1, 2], [3, 4, 5]])

>>> b
array([[0, 1, 2],
       [3, 4, 5]])

>>> b.ndim
2
```

- a = np.array([3., 4., 5.])

```
>>> a.ndim
1
```

Vetor NumPy: atributos

Dimensões (tamanho do array em cada eixo)

- Linhas por colunas

```
>>> b
    array([[0, 1, 2],
    [3, 4, 5]])

>>> b.shape
    (2, 3)
```

Vetor NumPy: atributos

Tamanho (*size*) e tipo (*dtype*)

- a.size? b.size?
- a.dtype?
- b.dtype?

Criando um vetor com valores

```
>>> c = np.arange(12).reshape(3,4)
```

```
>>> c
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

```
>>> c.size
12

>>> c.shape
(3, 4)

>>> c.ndim
2
```

Vetor/Matriz preenchidos

```
>>> np.zeros((1,1))

>>> np.zeros((1,2))

>>> np.zeros((2,2))

>>> np.zeros((2,3))

>>> np.zeros((3,2))
```

Vetor/Matriz preenchidos

- ZEROS cria um vetor e o preenche com 0 real
- ONES cria um vetor e o preenche com 1 real
- um = **np.ones((2,3))**
- Como visto, podemos mudar os parâmetros de criação dos vetores (próximo slide)

Vetor/Matriz preenchidos

Números aleatórios:

```
>>> randV = np.random.rand(2,3)

>>> randV
array([[0.91583606, 0.79393319, 0.07190635],
       [0.74404009, 0.81750605, 0.82854791]])
```

Vetor/Matriz preenchidos

Vazio. Cuidado!!!

```
>>> vazio = np.empty((3,3))


>>> vazio
array([[-1.28822975e-231, -1.28822975e-231,  2.47032823e-323],
       [ 0.00000000e+000,  0.00000000e+000,  0.00000000e+000],
       [ 0.00000000e+000,  0.00000000e+000,  0.00000000e+000]])
```

Vetor/Matriz preenchidos

Outras opções:

- Criar um *array* preenchido com outros números (!=0 && !=1)
  - np.full((2,3), 7)

- *Array* com intervalo espaçado
  - np.arange(0,10,2)

    vs.
  - np.linspace(0,10,2)

Salvando um vetor em arquivo

```
a = np.arange(0,3,.3).reshape(5,2)


>>> a
array([[0. , 0.3],
       [0.6, 0.9],
       [1.2, 1.5],
       [1.8, 2.1],
       [2.4, 2.7]])
```

Salvando um vetor em arquivo

```
a = np.arange(0,3,.3).reshape(5,2)

np.savetxt("saida.txt", a, delimiter=",")
```

Lendo de um arquivo

Vamos ver o arquivo “teste.csv”:

- np.loadtxt(“teste.csv”)
  - ?
- Argumentos da função loadtxt:
  - skiprows: recebe um inteiro com o número de linhas desconsideradas
  - unpack: se True, cada COLUNA é um *array* distinto. VERIFIQUEM A DIFERENÇA ENTRE unpack = True e unpack = False!
  - delimiter: define o caracter delimitador
  - dtype: especifica o tipo de dado (confira tabela com tipos)

Lendo de um arquivo

Vamos ler o arquivo “teste2.csv”:

- ?

Lendo de um arquivo

Vamos ler o arquivo “teste2.csv”:

- np.genfromtxt(“teste2.csv”)
  - Remover o cabeçalho: skip_header=N
  - genfromtxt converte *strings* em colunas numéricas para NaN!
  - Argumentos para lidar com valores faltantes:
    - filling_values = <x>: converte valores faltantes para o definido em <x>
    - missing_values: especifica o que são os valores faltantes

Datasets para exercício

- https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white.csv
- https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv
- Obtendo datasets via linha de comando:
  - cd <diretório de dados>
  - wget <URL>

Exercício

Dados os dois arquivos (vinho branco e tinto):

- Escrever um programa que **abra** e **leia** os arquivos
- **Troque** o “;” por “,”
- **Escreva** um arquivo contendo ambos *datasets*

- Lembrete: Só pode haver **um cabeçalho!**

Exercício

- Dado exercício anterior:
  - **Adicione** um rótulo como atributo final *(branco* ou *tinto)*
  - **Escreva** um arquivo contendo ambos *datasets*

- Lembrete: Só pode haver **um cabeçalho!**

O módulo CSV

import csv

- Solução (parcial):
  - import csv
  - F = open('winequality-red.csv')
  - tintos = list(csv.reader(F, delimiter=';'))

Exercício

Calcular a qualidade média dos vinhos

- Brancos
- Tintos
- Ambos

Exercício

Calcular a qualidade média dos vinhos

- Brancos, tintos e ambos

Como fazer?

- Extraia o último elemento de cada linha (menos cabeçalho)
- Converta cada elemento para o formato correspondente
- Insira os elementos em uma lista
- Divida a soma de todos os elementos pelo total de elementos

Exercício

Como fazer?

- Extraia o último elemento de cada linha (menos cabeçalho)
- Converta cada elemento para o formato correspondente
- Insira os elementos em uma lista
  - listaQuali = [float(elem[-1]) for elem in vinhos[1:]]

- Divida a soma de todos os elementos pelo total de elementos
  - sum(listaQuali) / len(listaQuali)

Exercício

Meia hora para fazer e entregar no Moodle!

Operações básicas em np.*arrays*

Elemento a elemento:

- /, *, +, -

- Vamos testar com um subconjunto (3 linhas – cabeçalho) dos *arrays* de vinhos branco e tinto.
  - Fazer as operações…

- E o produto matricial?

Operações básicas em np.*arrays*

Produto matricial:

- Dadas duas matrizes *numpy* A e B
  - A.dot(B)

    ou
  - np.dot(A, B)

Operações básicas em np.*arrays*

- Numpy média e desvio padrão?
  - np.mean()
  - np.std()

- Como fazer para calcular de uma matriz inteira?
- E de apenas uma coluna/eixo?

- https://docs.scipy.org/doc/numpy/reference/generated/numpy.mean.html
- https://docs.scipy.org/doc/numpy/reference/generated/numpy.std.html

Exercício

Dados os dois arquivos de vinhos:

- Calcular a média e o desvio padrão de cada um dos atributos para cada um dos tipos de vinho
- Calcular para o arquivo contendo ambos os vinhos
- Imprimir os dados organizadamente

Exercício

- Ler e tratar o arquivo “taubaM.txt”
  - Qual a temperatura máxima e mínima do ano?
  - Qual a média de temperatura anual?
  - Quantas horas de sol foram contabilizadas durante o ano?
  - Qual mês teve mais dias com chuvas maiores que 1mm
  - Quantos mm de chuva foram contabilizados durante o ano?
  - Qual a média de mm de chuva por mês?