

DEEP LEARNING BRASIL

SUMMER SCHOOL

Python warm up

Marcelo Piovan

piovan@heurys.com.br

Data H

Patrocínio:



www.deeplearningbrasil.com.br



Apoio:



Agenda

- O que é Python?
- Montando um ambiente do zero
 - Instalando o Python
 - Instalando e configurando o Virtualenv
 - Instalando pacotes
- Fundamentos da linguagem
- Numpy
- Scikit

O que é Python?

- Python é uma linguagem de propósito geral de alto nível, **multi paradigma**, suporta o paradigma orientado a objetos, funcional e procedural. Possui tipagem dinâmica e uma de suas principais características é permitir a fácil leitura do código e exigir poucas linhas de código se comparado ao mesmo programa em outras linguagens.

Fonte: <https://pt.wikipedia.org/wiki/Python>

Qual versão utilizar ?

- Atualmente o Python possui duas versões 2.7 e 3.7
- Deadline para a versão 2.7 ~2020

<https://pythonclock.org/>

Fonte: <https://wiki.python.org/moin/Python2orPython3>

DEEP LEARNING BRASIL

SUMMER SCHOOL

Montando um ambiente do zero

Patrocínio:



www.deeplearningbrasil.com.br

Apoio:



Instalando o Python

- Linux (Ubuntu 16.04)

```
$ sudo apt-get install software-properties-common  
$ sudo add-apt-repository ppa:deadsnakes/ppa  
$ sudo apt-get update  
$ sudo apt-get install python
```

- Windows

- Efetuar o download da versão e executar o instalador
 - <https://www.python.org/ftp/python/3.5.4/python-3.5.4-amd64.exe>
 - <https://www.python.org/ftp/python/3.5.4/python-3.5.4.exe>

Verificando a instalação do Python

- Linux

`$ python3`

- Windows

`C:\<diretório de instalação>\python.exe`

```
Python 3.5.2 (default, Nov 23 2017, 16:37:01)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Instalando o gerenciador de pacotes do Python (PIP)

- Fazer o download do script de instalação (get-pip.py)
 - <https://bootstrap.pypa.io/get-pip.py>
- Executar o script com o Python
 - Linux
 - \$ `python3 get-pip.py`
 - Windows
 - C:\<diretório de instalação>\python.exe get-pip.py

O que é o VirtualEnv ?

- É uma ferramenta para criar ambientes Python isolados.

Instalando o VirtualEnv

- Linux

```
$ sudo pip3 install virtualenv
```

- Windows

```
C:\<diretório de instalação>\Scripts\pip.exe install virtualenv
```

Utilizando o VirtualEnv

- Criando um ambiente virtual

- Linux

```
$ virtualenv -p <python> <diretório_destino>
```

- Windows

```
C:\<diretório de instalação>\Scripts\virtualenv.exe <diretório_destino>
```

Ativando o VirtualEnv

- Ativando o ambiente virtual

- Linux

`$ source <diretório_destino>/bin/activate`

- Windows

`<diretório_destino>\Scripts\activate`

```
C:>c:\dados\python\virtualEnvs\summerschool\Scripts\activate  
(summerschool) C:>
```

Desativando o VirtualEnv

- Linux

`(virtualenv) $ deactivate`

- Windows

`(virtualenv) C:\deactivate`

Instalando pacotes

- Scikit-learn : É uma ferramenta simples e eficiente para data mining e data analysis.

- Linux

`(virtualenv) $ pip install scikit-learn`

- Windows

`(virtualenv) pip install scikit-learn`

Instalando pacotes

- Numpy: Numpy é a biblioteca para computação científica em python. É um pacote que suporta arrays e matrizes multidimensionais, possuindo uma larga coleção de funções matemáticas para trabalhar com estas estruturas.

- Linux

`(virtualenv) $ pip install numpy`

- Windows

`(virtualenv) pip install numpy`

Instalando pacotes

- Matplotlib: É uma biblioteca para gerar gráficos 2D.

- Linux

`(virtualenv) $ pip install matplotlib`

- Windows

`(virtualenv) pip install matplotlib`

DEEP LEARNING BRASIL

SUMMER SCHOOL

Fundamentos da linguagem Python

Patrocínio:



www.deeplearningbrasil.com.br



Apoio:



Convenções

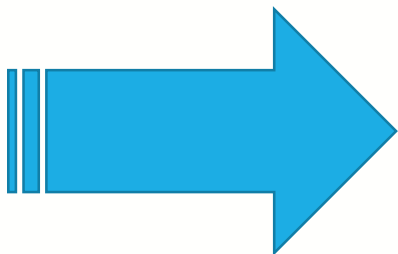
- Métodos, variáveis e funções devem utilizar o padrão Snake-case, onde os nomes são sempre em minúsculos separados por um underscore. Exemplos:
 - `dia_do_nascimento = 20`
 - `def gerar_nota_fiscal(self):`
- Classes devem começar com uma letra maiúscula e as palavras são separadas por letra maiúscula. Exemplo:
 - `class PessoaFisica(object):`
- Os comandos dentro de um bloco é definido por sua indentação, geralmente utilizamos 2 espaços ou um tab.

Declaração de variáveis

- Em Python não existe uma declaração explícita de variáveis, ou seja, ela só passa a existir quando atribuímos um valor a variável.
 - nome_usuario = "piovan"
 - email_usuario = 'piovan@heurys.com.br'
- As variáveis tem seu tipo dinâmico, conforme são atribuídas.

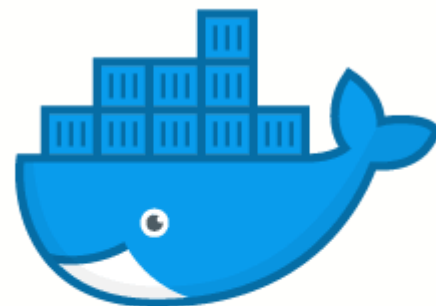
Vamos ver alguns exemplos !

- Declaração de variáveis
- Precisão do tipo de dados float



<https://colab.research.google.com>

<https://goo.gl/svtXdW>



docker

Operadores

- Temos várias categorias de operadores em Python
 - Aritméticos
 - Lógicos
 - Atribuições
 - Bitwise

Operadores Aritméticos

Operador	Exemplo (a=10, b=15)	Descrição
+	$a + b = 25$	Adição
-	$a - b = -5$	Subtração
*	$a * b = 150$	Multiplicação
/	$a / b = 0.666666666$	Divisão
%	$a \% b = 10$	Módulo
**	$a ** b = 10000000000000000$	Potência
//	$4 // 3 = 1$	Divisão (resultado inteiro)

Operadores Lógicos

Operador	Exemplo (a=10, b=15)	Descrição
==	a == b = False	Igual
!=	a != b = True	Diferente
<>	a <> b = True	Diferente
>	a > b = False	Maior
<	a < b = True	Menor
>=	a >= b = False	Maior ou igual
<=	a <= b = True	Menor ou igual
in	a in [10,20,30] = True	Esta em
not in	a not in [10,20,30] = False	Não esta em
is	a is 10 = True	É
not is	a is not 10 = False	Não é

Operadores para Atribuições

Operador	Exemplo (a=10, b=15)
=	a = 10 > 10
+=	a += 1 > 11
-=	a -= 1 > 10
*=	a *= 2 > 20
/=	a /= 2 > 10.0
%=	a %= 4 > 2.0
**=	a **= 4 > 16.0
//=	a //= 3 > 5.0

Operadores Bitwise

Operador	Exemplo (a=10, b=15)	Descrição
&	$a \& b = 10$	Binary AND
	$a b = 15$	Binary OR
^	$a \wedge b = 5$	Binary XOR
~	$(\sim a) = -11$	Binary Ones
<<	$a \ll 1 = 20$	Binary Left Shift
>>	$a \gg 1 = 5$	Binary Right Shift

Estrutura do comando if

if <condição>:

código ...

elif <condição>:

código ...

else:

código ...

List, Tuple, Dictionary, Set

- **[]** - **List**, permite criar uma lista **mutável** de várias dimensões.
- **()** - **Tuple**, permite criar uma lista **imutável** de várias dimensões.
- **{ }** - **Dictionary**, permite criar estrutura chave, valor
- **{ }** – **set**, permite criar conjuntos com valores que não repetem.

List

- Declaração: lista = [1, 2, 3, ...]

Métodos	Descrição
append(valor)	Adiciona um valor no fim da lista
pop()	Remove o último item da lista
remove(valor)	Remove um valor da lista
index(valor)	Retorna a posição do valor na lista
sort()	Ordena a lista
insert(índice, valor)	Inclui o valor em uma posição da lista
reverse()	Inverte uma lista
len(list tuple)	Retorna a quantidade de elementos da lista

Tuple

- Declaração: tupla = (1, 2, 3, ...)

Métodos	Descrição
index(valor)	Retorna a posição do valor na tupla
len(list tuple)	Retorna a quantidade de elementos da tupla

Dictionary

- Declaração: dicionário = { 'chave_1': 1, 'chave_2': 2 }

Métodos	Descrição
items()	Retorna a lista como uma lista
clear()	Limpa a lista
get(key)	Recupera o valor da chave
keys()	Retorna uma lista com as chaves
values()	Retorna uma lista com os valores
pop(key)	Exclui um nó da estrutura
copy()	Copia uma estrutura

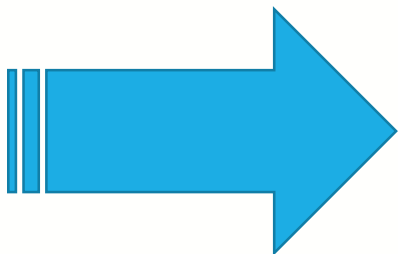
Set

- Declaração: conjunto = {'a', 'b', 'c', 'd'}

Métodos	Descrição
add(valor)	Adiciona um valor ao conjunto
remove(valor)	Remove um valor do conjunto
set(valor)	Cria o conjunto
pop()	Remove o primeiro item do conjunto
isdisjoint(valor)	Retorna se um item existe no conjunto
clear()	Limpa um conjunto
copy()	Copia um conjunto

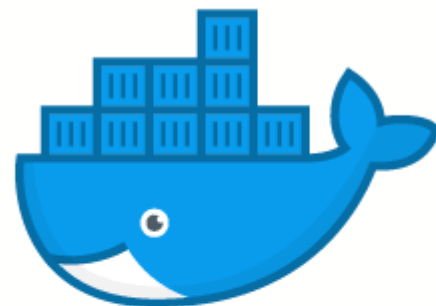
Vamos ver alguns exemplos !

- List, Tuple, Dictionary, Set



<https://colab.research.google.com>

<https://goo.gl/svtXdW>



docker

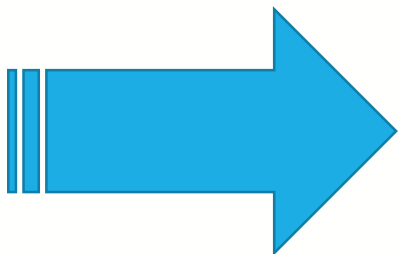
Percorrendo as estrutura de dados

- Para percorrer os itens de um list, tuple, set ou dictionary utilizamos o comando for in, veja a sintaxe abaixo:

for item **in** list | tuple | set | dictionary

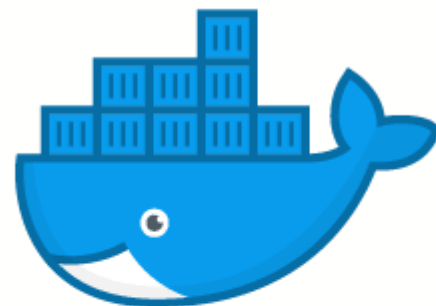
Vamos ver alguns exemplos !

- For in, zip, len, range, matriz



<https://colab.research.google.com>

<https://goo.gl/svtXdW>



docker

Definindo funções

- Uma função é um bloco de código organizado e reutilizável que é usado para executar uma **única ação** relacionada.

```
def <nome>(<parametros>):
```

Código...

Definindo função lambda

- O que é função Lambda?

Lambda são funções anônimas que retornam um único valor ou expressão.

lambda <parametros> : expressão

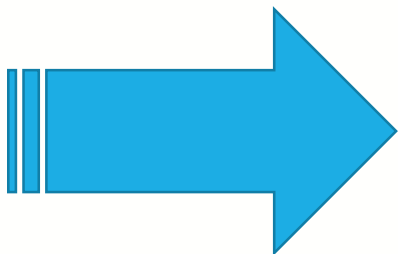
Percorrendo uma estrutura com o map

- map: Aplica uma função a cada item de uma estrutura iterável.

map(<função>, <objeto iterável>)

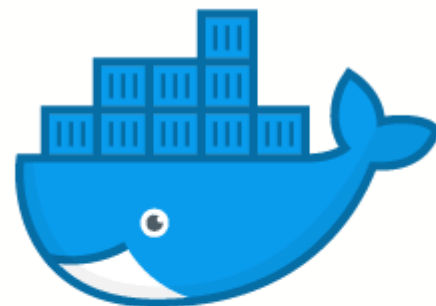
Vamos ver alguns exemplos !

- Definição de função no python



<https://colab.research.google.com>

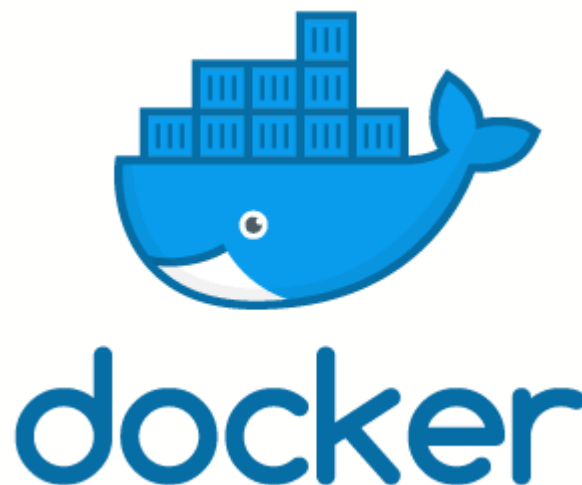
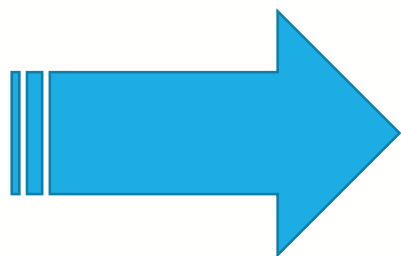
<https://goo.gl/svtXdW>



docker

Vamos praticar !

- Exercicio_1.ipynb



DEEP LEARNING BRASIL

SUMMER SCHOOL



Numpy

Patrocínio:

ZG
SOLUÇÕES

Saúde**mobi** 
INFOMACH
TECNOLOGIA PARA NEGÓCIOS

 INSTITUTO DE
INFORMÁTICA
UFPA

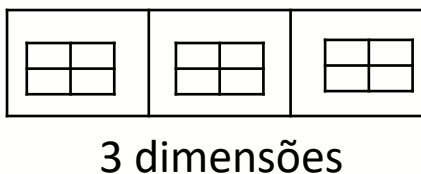
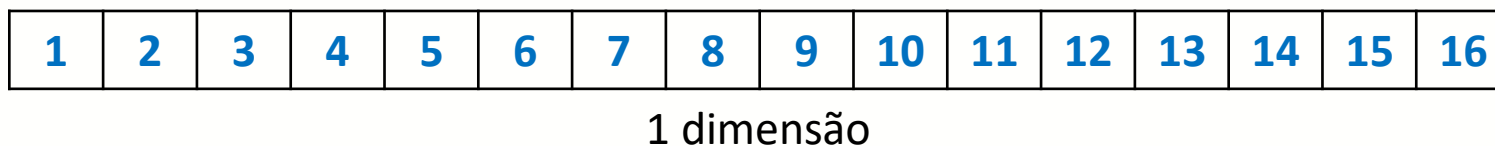
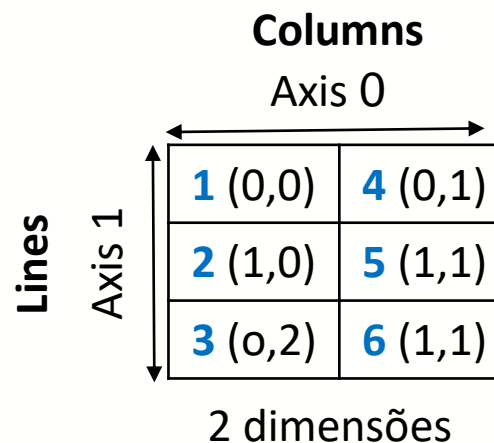
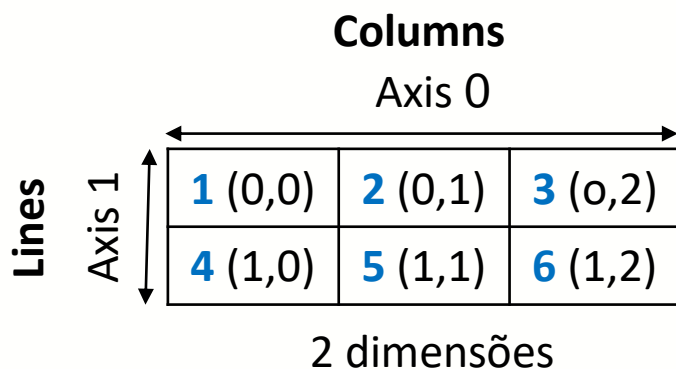
 **FASAM**
FACULDADE SUL-AMERICANA

 www.deeplearningbrasil.com.br

Apoio:

 **NVIDIA** 

Criando arrays com Numpy



Criando Arrays com Numpy

- A API do numpy nos permite:
 - Criar arrays com várias dimensões (**array**)
 - Criar arrays com dados sequenciais (**arange**)
 - Criar arrays randômicos entre 0-1 (**empty**), randômicos (**random**), inicializados com 1 (**ones**), inicializados com um número (**full**), inicializados com 0 (**zeros**)
 - Criar arrays bidimensionais diagonais (**diag**) ou de identidade (**eye**)
 - Saber o formato (**shape**) de um array, o tamanho(**size**), a dimensão (**ndim**) e o tipo dos elementos (**dtype**).

Transformando os arrays com Numpy

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----

1 x 16

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

4 x 4

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16

2 x 8

1	2
3	4
5	6
7	8
9	10
11	12
13	14
15	16

8 x 2

6	7
10	11

2 x 2

4	8	12	16
---	---	----	----

1 x 4

Transformando os arrays com Numpy

- A API do numpy nos permite:
 - Transformar um array em outro formato (**reshape**, **resize**)
 - Transformar um array de várias dimensões em um array de uma dimensão (**ravel**)
 - Ordenar os elementos do array no padrão C (linha) ou Fortran (coluna)
 - Extrair partes de um array (**slice [line:column]**)
 - Adicionar uma nova dimensão (**newaxis**)

Transformando os arrays com Numpy

Matriz 1

1	2	3
4	5	6
7	8	9

Concatenar

Matriz 2

1	1	1
2	2	2
3	3	3

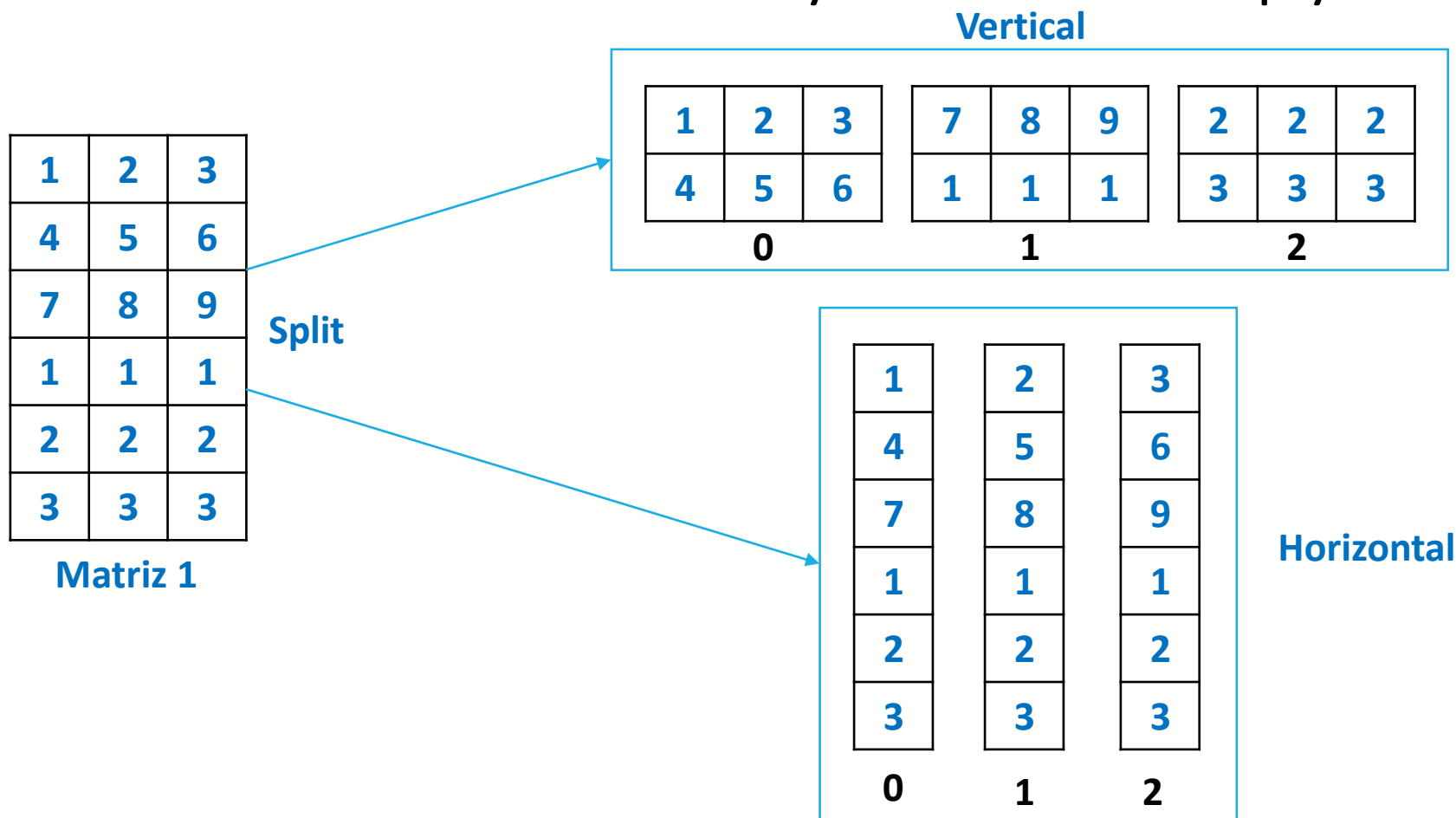
1	2	3
4	5	6
7	8	9
1	1	1
2	2	2
3	3	3

Vertical

1	2	3	1	1	1
4	5	6	2	2	2
7	8	9	3	3	3

Horizontal

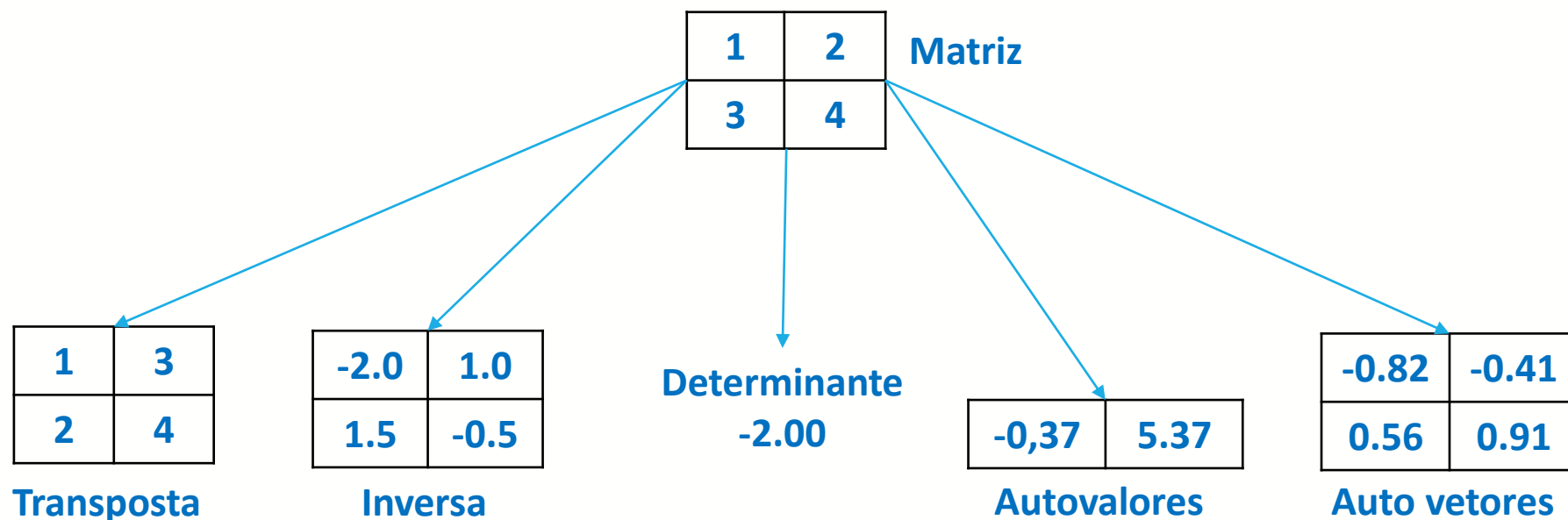
Transformando os arrays com Numpy



Transformando os arrays com Numpy

- A API do numpy nos permite:
 - Concatenar na vertical (**concatenate**, **vstack**)
 - Concatenar na horizontal (**hstack**)
 - Dividir na vertical (**split**, **vsplit**)
 - Dividir na horizontal (**hsplit**)

Aplicando cálculos nos arrays com Numpy



soma, soma da diagonal, maior, menor, raiz, seno, cosseno, tangente,
Exponencial e log dos elementos do array etc...

Aplicando cálculos nos arrays com Numpy

- A API do numpy nos permite:
 - Calcular a matriz transposta de uma matriz (**T**)
 - Calcular a matriz inversa de uma matriz (**linalg.inv**)
 - Calcular a determinante da matriz (**linalg.det**)
 - Calcular os auto vetores e autovalores de uma matriz (**linalg.eig**)
 - Somar a diagonal de um array com mais de uma dimensão (**trace**)
 - Somar um array (**sum**), encontrar o maior elemento (**max**), encontrar o menor elemento (**min**).
 - Calcular a raiz (**sqrt**), desvio padrão (**std**), seno (**sin**), cosseno (**cos**) e tangente (**tan**) dos elementos do array
 - Calcular o exponencial (**exp**) e o logaritmo (**log**) de um array.

Aplicando cálculos nos arrays com Numpy

1	2	3
4	5	6
7	8	9

Matriz A

+

1	2	3
4	5	6
7	8	9

Matriz B

=

2	4	6
8	10	12
14	16	18

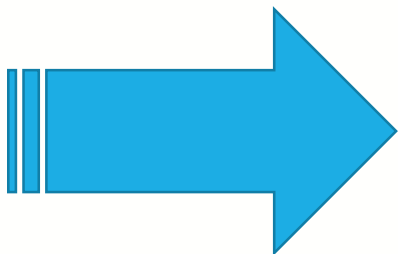
+ - * / > <

Aplicando cálculos nos arrays com Numpy

- A API do numpy nos permite:
 - Somar, subtrair, multiplicar, dividir um array por outro array
 - Aplicar uma expressão/constante sobre um array
 - Aplicar expressões booleanas

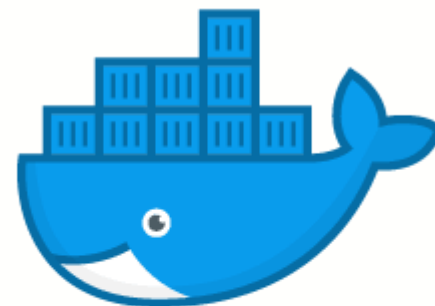
Vamos ver alguns exemplos !

- Usando o Numpy



<https://colab.research.google.com>

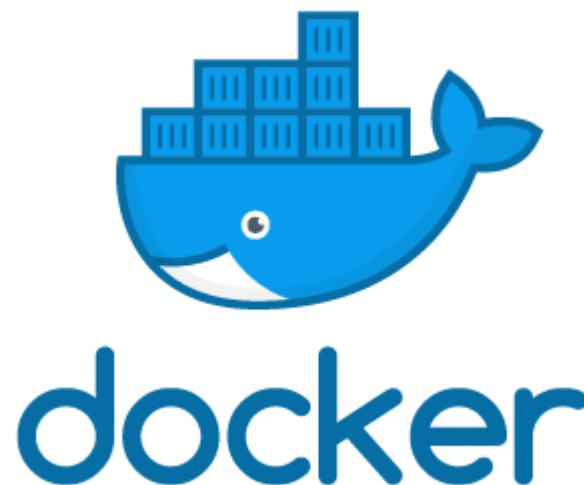
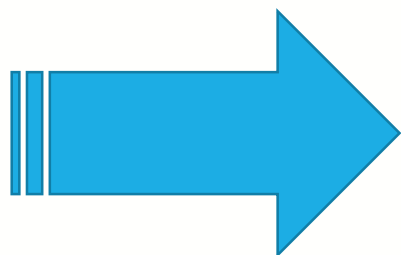
<https://goo.gl/qT7LwV>



docker

Vamos praticar !

- Exercicio_2.ipynb



DEEP LEARNING BRASIL

SUMMER SCHOOL



Scikit

Patrocínio:


ZG
SOLUÇÕES

Saúdemobi

INFOMACH
TECNOLOGIA PARA NEGÓCIOS

 **INSTITUTO DE
INFORMÁTICA**
UFPA

 **FASAM**
FACULDADE SUL-AMERICANA

 www.deeplearningbrasil.com.br

Apoio:

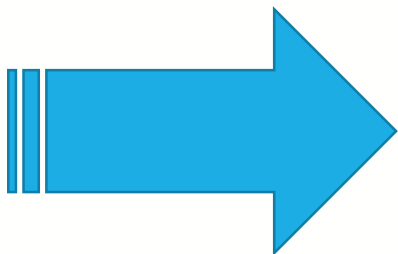
 **NVIDIA** **DATA^H**

O que é Scikit ?

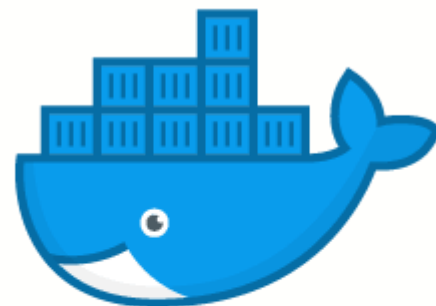
- É uma biblioteca de aprendizado de máquina de código aberto para a linguagem de programação Python. Ela inclui vários algoritmos de classificação, regressão e agrupamento entre outros.

Fazendo previsões com Scikit

- Usando a regressão linear simples para prever.



<https://colab.research.google.com>



docker

