Análisis de Datos y Aprendizaje Máquina con Tensorflow 2.0: Pre-procesamiento de Datos para Aprendizaje Máquina

2019/09/30

Álgebra Lineal

Objetivo: El núcleo de deep learning son las operaciones de matrices. Se conocerán las formas de crear y manipular estas estructuras.

Notación Básica

Vector como lista

```
In [1]: v1 = [2, 4, 6]
        v1

Out[1]: [2, 4, 6]
```

Lista a arreglo de numpy

```
In [2]: import numpy as np
        v2 = np.array([1, 3, 5])
        v2

Out[2]: array([1, 3, 5])

In [3]: v3 = np.arange(1, 8)
        v3

Out[3]: array([1, 2, 3, 4, 5, 6, 7])
```

Creación de matrices

Matriz de ceros 2x2

```
In [4]: np.zeros((2,2))

Out[4]: array([[0., 0.],
               [0., 0.]])
```

Matriz de unos 2x2

```
In [5]: np.ones((2,2))

Out[5]: array([[1., 1.],
               [1., 1.]])
```

Matriz identidad 3x3

```
In [6]: np.eye(3)

Out[6]: array([[1., 0., 0.],
               [0., 1., 0.],
               [0., 0., 1.]])
```

Matriz random

```
In [7]: np.random.rand(3,3)

Out[7]: array([[0.60915219, 0.42782664, 0.50621834],
               [0.06659117, 0.91469226, 0.01323167],
               [0.55522198, 0.62979646, 0.23764036]])
```

Matriz sin inicializar entradas

```
In [8]: np.empty((3,3))

Out[8]: array([[0.60915219, 0.42782664, 0.50621834],
               [0.06659117, 0.91469226, 0.01323167],
               [0.55522198, 0.62979646, 0.23764036]])
```

Números espaceados

```
In [9]: #entre 1 y 5
        print(np.linspace(1,5,21))

[1.  1.2 1.4 1.6 1.8 2.  2.2 2.4 2.6 2.8 3.  3.2 3.4 3.6 3.8 4.  4.2 4.4
 4.6 4.8 5. ]
```

Indexing

Acceso a un elemeto del vector

```
In [10]: v4 = np.array([1, 3, 5, 9])
         v4[2]

Out[10]: 5
```

Acceso a un elemento de la matriz

```
In [11]: A = np.array([[9, 3, 2],
                       [1, 0, 0],
                       [1, 2, 2]])
         A[0,0]
```

```
Out[11]: 9
```

Todos los elementos de la fila 1

```
In [12]: A[0,:]

Out[12]: array([9, 3, 2])
```

Todos los elementos de la columna 1

```
In [13]: A[:,0]

Out[13]: array([9, 1, 1])
```

Elementos 1 a 2 de la fila 2

```
In [14]: A[1, 0:2]

Out[14]: array([1, 0])
```

Elementos 2 a 3 de la columna 1

```
In [15]: A[1:3, 0]

Out[15]: array([1, 1])
```

Elementos a partir del renglon 2

```
In [16]: A[1:, :]

Out[16]: array([[1, 0, 0],
                [1, 2, 2]])
```

Tamaño matriz

```
In [17]: np.size(A)

Out[17]: 9
```

Número renglones y columnas

```
In [18]: B = np.array([[9, 3, 2],
                       [1, 8, 2]])

         np.size(B, 0), np.size(B, 1)

Out[18]: (2, 3)
```

Crear matriz de tamaño m

```
In [19]: np.zeros(np.size(B))

Out[19]: array([0., 0., 0., 0., 0., 0.])
```

Operaciones simples con vectores y matrices

Operaciones con elementos

```
In [20]: a = np.array([1, 3, 5, 9])
         2*a
```

```
Out[20]: array([ 2,  6, 10, 18])
```

```
In [21]: a/2
```

```
Out[21]: array([0.5, 1.5, 2.5, 4.5])
```

```
In [22]: b = np.array([8, 4, 5, 1])
         a + b, a - b
```

```
Out[22]: (array([ 9,  7, 10, 10]), array([-7, -1,  0,  8]))
```

```
In [23]: np.power(a, 2)
```

```
Out[23]: array([ 1,  9, 25, 81])
```

```
In [24]: a*b
```

```
Out[24]: array([ 8, 12, 25,  9])
```

```
In [25]: a/b
```

```
Out[25]: array([0.125, 0.75 , 1.   , 9.   ])
```

Operaciones con vectores

```
In [26]: a = np.array([1, 4, 6, 3])
         np.sum(a), np.mean(a), np.var(a), np.std(a)
```

```
Out[26]: (14, 3.5, 3.25, 1.8027756377319946)
```

```
In [27]: np.max(a), np.min(a)
```

```
Out[27]: (6, 1)
```

Si se da una matriz, el segundo argumento indica la dimension sobre que se hará la operación

```
In [28]: a = np.array([[9, 3, 2],
                       [4, 5, 6]])
         np.mean(a, 1), np.mean(a, 0)
```

```
Out[28]: (array([4.66666667, 5.        ]), array([6.5, 4. , 4. ]))
```

Vector fila 1x3 por 3x1 vector columna

```
In [29]: a = np.array([[9, 3, 2]])
         b = np.array([[1, 7, 5]])
         np.matmul(a, b.T)
```

```
Out[29]: array([[40]])
```

Vector fila 3x1 por 1x3 vector columna

```
In [30]: np.matmul(a.T, b)
```

```
Out[30]: array([[ 9, 63, 45],
                [ 3, 21, 15],
                [ 2, 14, 10]])
```

Operaciones con Matrices

```
In [31]: a = np.random.rand(3,2)
         b = np.random.rand(2,4)
         c = np.matmul(a, b)
         c

Out[31]: array([[0.1973519 , 0.22038879, 0.09369738, 0.27751112],
                [0.23615492, 0.69708248, 0.15387783, 0.79138184],
                [0.70891482, 1.16078866, 0.37214159, 1.3880795 ]])

In [32]: a = np.random.rand(3,2)
         b = np.random.rand(1,3)
         c = np.matmul(b, a)
         c

Out[32]: array([[0.46396768, 0.68224367]])

In [33]: a = np.array([[9, 3, 2],
                       [1, 0, 0],
                       [1, 2, 2]])

         np.linalg.inv(a)

Out[33]: array([[-2.46716228e-17,  1.00000000e+00,  2.46716228e-17],
                [ 1.00000000e+00, -8.00000000e+00, -1.00000000e+00],
                [-1.00000000e+00,  7.50000000e+00,  1.50000000e+00]])

In [34]: evals, evecs = np.linalg.eig(a)
         evals, evecs

Out[34]: (array([ 9.62822934, -0.1376201 ,  1.50939076]),
          array([[-0.98254145, -0.10320868, -0.20455998],
                 [-0.10204799,  0.7499535 , -0.13552487],
                 [-0.1555587 , -0.6533894 ,  0.96942675]]))
```

Cambiar forma de matrices

```
In [35]: a = np.array([[9, 3],
                       [1, 0],
                       [1, 2]])

         b = np.reshape(a, (6,1))
         b

Out[35]: array([[9],
                [3],
                [1],
                [0],
                [1],
                [2]])
```

```
In [36]: a.sum()

Out[36]: 16

In [37]: np.reshape(a, (2,3))

Out[37]: array([[9, 3, 1],
                [0, 1, 2]])

In [38]: a = np.array([[1, 2], [3, 4]])
         b = np.array([[5, 6]])
         np.concatenate((a, b), axis=0)

Out[38]: array([[1, 2],
                [3, 4],
                [5, 6]])
```

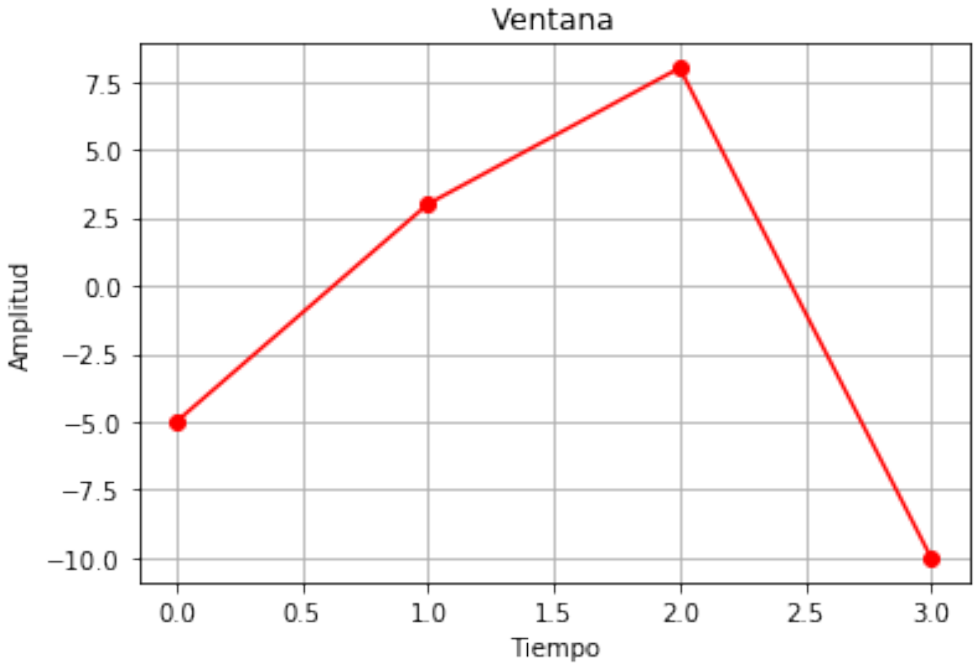
Ciclos

```
In [39]: a = np.array([-5, 3, 8, -10, 11, -7])
         for i in a:
             if(i > 10):
                 print("Mayor que 10")
             if(i < 0):
                 print("Valor Negativo")
             else:
                 print('Else')

Valor Negativo
Else
Else
Valor Negativo
Mayor que 10
Else
Valor Negativo
```

Plotear Vector

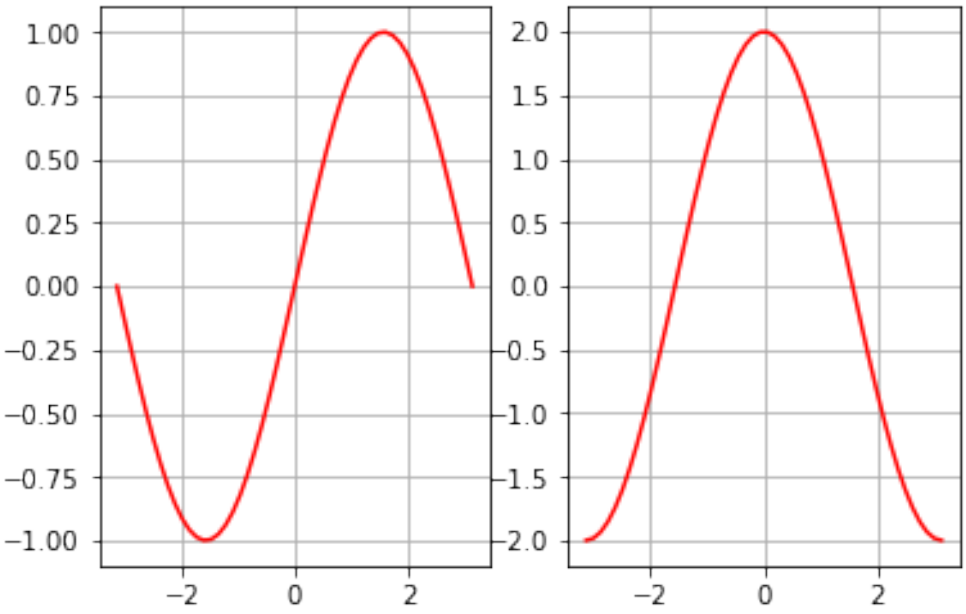
```
In [40]: %matplotlib inline
         import matplotlib.pyplot as plt
         import matplotlib as mpl

         x1 = np.array([-5, 3, 8, -10])

         fig = plt.figure(figsize=(6,4))
         ax = fig.add_subplot(1, 1, 1)
         ax.plot(x1, "-ro")
         ax.set_xlabel('Tiempo')
         ax.set_ylabel('Amplitud')
         ax.set_title('Ventana')
```

```
plt.grid (True)
plt.show()
```



Plotear Función

```
In [41]: x2 = np.linspace(-np.pi, np.pi)
         fig = plt.figure(figsize=(6,4))
         ax = fig.add_subplot(1, 2, 1)
         plt.plot(x2, np.sin(x2), "-r")
         plt.grid (True)
         ax = fig.add_subplot(1, 2, 2)
         plt.plot(x2, 2*np.cos(x2), "-r")
         plt.grid (True)
         plt.show()
```



In [42]:
```
a = np.random.rand(64,64)
plt.figure(figsize=(6,4))
imgplot = plt.imshow(a)
```