Código Aprendizaje automático

Machine Learning Code

Autor: John Edward Ospina Ladino
IS&C, Universidad Tecnológica de Pereira, Pereira, Colombia
Correo-e: j.ospina12@utp.edu.co

***Resumen*— Este documento presenta un resumen explicativo del código de machine learning presentado en la materia de computación blanda, donde utilizamos la librería de python numpy, al inicio del mismo se presente una explicación del uso básico de los arreglos con dicha librería, para finalmente presentar un problema de tipo real en el cual se podría utilizar esta misma librería para dar solución a dicho problema**

***Palabras clave*— Machine learning, algorithm, proceso, función,arreglo,librería,conjunto, matriz, vector.**

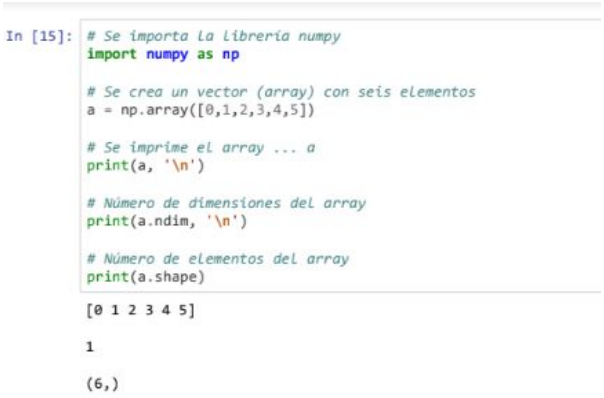
Abstract— This document presents an explanatory summary of the machine learning code presented in the matter of soft computing, where we use the python numpy library, at the beginning of it an explanation of the basic use of the arrays with said library is presented, to finally present a real type problem in which this same library could be used to solve this problem

Keywords— Machine learning, algorithm, process, function, array, library, set, matrix, vector.

I. INTRODUCCIÓN

El machine learning ha sido categorizado como uno de los tópicos más importantes en la ciencia de la computación, este campo promete un gran futuro en el mundo de la inteligencia artificial proponiendonos escenarios que tal vez en algún momento la humanidad creía imposibles de llegar.

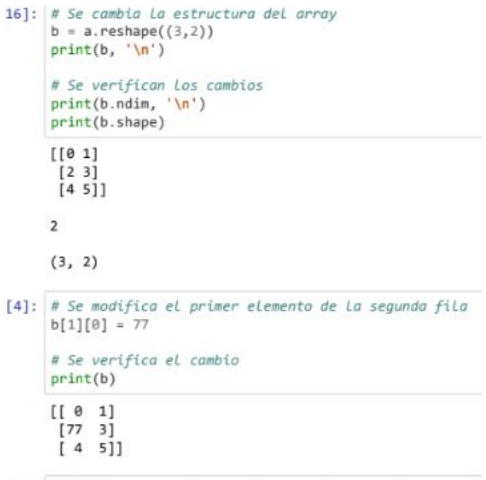
En las siguientes líneas de este documento se presentará un ejemplo real donde se trata de explicar de manera concisa y precisa el uso del machine learning a través del lenguaje de programación python y su librería numpy.



```
In [15]: # Se importa la librería numpy
         import numpy as np

         # Se crea un vector (array) con seis elementos
         a = np.array([0,1,2,3,4,5])

         # Se imprime el array ... a
         print(a, '\n')

         # Número de dimensiones del array
         print(a.ndim, '\n')

         # Número de elementos del array
         print(a.shape)

         [0 1 2 3 4 5]

         1

         (6,)
```

Figura 1: Machine Learning 01 parte 1

Inicialmente se presenta la importación de la librería numpy la cual se debe inicializar con un nombre que puede ser cualquiera, se inicia asignando un nuevo array de 0 a 5 a la variable a para posteriormente imprimir este mismo y probar las funciones, ndim que nos permite ver las dimensiones del array, y shape que nos arroja el número de elementos en el mismo.



```
16]: # Se cambia la estructura del array
     b = a.reshape((3,2))
     print(b, '\n')

     # Se verifican los cambios
     print(b.ndim, '\n')
     print(b.shape)

     [[0 1]
      [2 3]
      [4 5]]

     2

     (3, 2)

[4]: # Se modifica el primer elemento de la segunda fila
     b[1][0] = 77

     # Se verifica el cambio
     print(b)

     [[ 0  1]
      [77  3]
      [ 4  5]]
```

Figura 2: Machine Learning 01 Parte 2

Se introduce una nueva función como lo es el reshape, que nos permite cambiar la forma de un arreglo, en este ejemplo volvemos un arreglo unidimensional, uno bidimensional,

adicionalmente se nos indica cómo hacer una alteración en una posición específica.

```
In [14]: # Se realiza una copia del array
         c = a.reshape((3,2)).copy()
         print(c, '\n')

         # Se cambia el primer valor de c
         c[0][0] = -99

         # El array a no se modifica
         print(a, '\n')

         # El array c queda modificado
         print(c)

[[ 0  1]
 [77  3]
 [ 4  5]]

[ 0  1 77  3  4  5]

[[-99   1]
 [ 77   3]
 [  4   5]]
```

Figura 3: Machine Learning 01 Parte 3

Se introduce y prueba la función copy que permite realizar una copia además se verifican nuevamente las funciones presentadas anteriormente.

```
In [17]: # Las operaciones se propagan a lo largo del array
         d = np.array([1,2,3,4,5])

         # Se multiplican los elementos por 2
         print(d*2, '\n')

         # Se elevan al cuadrado los elementos del array
         print(d**2)

[ 2  4  6  8 10]

[ 1  4  9 16 25]
```

Figura 4: Machine Learning 01 Parte 4

Comprobamos que al realizar una operación con un array, se afecta cada uno de sus elementos.

```
In [31]: # Nueva definición para el array a
         a = np.array([1,2,3,4,5])

         # Itera sobre todos los elementos del array
         print(a>4, '\n')

         # Se ejecuta la instrucción para los elementos que cumplan la condicion: "elem
         ento > 4"
         # Se cambian el contenido de los elmentos mayores a 4
         a[a>4] = 1
         print(a, '\n')

         # Los elementos cuyo contenido es igual a 1, reciben como nuevo valor el númer
         o 777
         a[a==1] = 777
         print(a, '\n')

[False False False False  True]

[1 2 3 4 1]

[777   2   3   4 777]
```

Figura 5: Machine Learning 01 Parte 5

Se inicia a realizar pruebas para evaluar elementos dentro de un arreglo

```
In [35]: # Control de valores erróneos
         c = np.array([1, 2, np.NAN, 3, 4])
         print(c, 'n')

         # Se verifica la existencia de valores nan
         print(np.isnan(c), '\n')

         # Se eligen todos los valores que NO son nan
         print(c[~np.isnan(c)], '\n')

         # Se calcula el promedio de los valores que NO son nan
         print(np.mean(c[~np.isnan(c)]))

[ 1.  2. nan  3.  4.] n
[False False  True False False]

[1. 2. 3. 4.]

2.5
```

Figura 6: Machine Learning 01 Parte 6

Se introduce un nuevo valor de dato, el .nan que se refiere básicamente a un valor erróneo, además se realiza la introducción de la función isnan para verificar si un elemento en específico es de este tipo.

```
In [ ]: # PROYECTO machine learning
        Enunciado
        # Enunciado: Una empresa vende el servicio de proporcionar algoritmos de apren
        dizaje automático a través de HTTP.
        # Con el éxito creciente de la empresa, aumenta la demanda de una mejor infrae
        structura para atender todas las
        # solicitudes web entrantes. No queremos asignar demasiados recursos, ya que s
        ería demasiado costoso.
        # Por otro lado, perderemos dinero si no hemos reservado suficientes recursos
         para atender todas las solicitudes entrantes.
        # Ahora, la pregunta es, ¿cuándo alcanzaremos el límite de nuestra infraestruc
        tura actual, que se estima en
        # 100.000 solicitudes por hora?. Nos gustaría saberlo de antemano cuando tenem
        os que solicitar servidores adicionales
        # en la nube para atender todas las solicitudes con éxito sin pagar por las no
        utilizadas.

In [50]: # Vamos a desarrollar un programa de machine learning (básico)
         # El siguiente es un paquete de datos a ser procesdos:
         # La primera columna es: Número de horas
         # La segunda columna es: Número de tareas ejecutadas
         data = np.genfromtxt("web_traffic.tsv", delimiter="\t")
         print(data[:10], '\n')

         # Número de datos
         print(data.shape)
```

Figura 7: Machine Learning 01 Parte 7

Se presenta el ejemplo para resolver con machine learning, y se procede a recibir el archivo.tsv que ayudará a la solución del problema debido a que en este se encuentran los datos a evaluar.

```
)                                                    machine learning 01

In [51]: # Se divide el array en dos vectores columnas: x, y
         x = data[:,0]
         y = data[:,1]

         # Se muestran los valores en x, y
         print(x, '\n')
         print(y, '\n')
```

Figura 8: Machine Learning 01 Parte 8

Se procede a separar el array en dos vectores x e y, posteriormente se imprimen

La impresión de ambos vectores es bastante larga y no tan significativa para la explicación del programa desarrollado en este documento.

```
In [52]: # Dimensión de los vectores x, y
         print(x.ndim, '\n')
         print(y.ndim, '\n')

         # Elementos contenidos en los vectores x, y
         print(x.shape, '\n')
         print(y.shape)

         1

         1

         (743,)

         (743,)

In [53]: # Investigamos el número de valores nan que contiene el vector y
         print(np.sum(np.isnan(y)))

         8
```

Figura 9: Machine Learning 01 Parte 9

En esta parte del código, se muestra la dimensión de cada uno de los arreglos, cada uno de dimensión 1 y el número de elementos en cada uno de ellos.

Posteriormente se investiga el número de valores “nan” que aparecen en el arreglo Y.

```
In [54]: # Número de elementos en x, y, antes de ser comprimidos
         print(x.shape, '\n')
         print(y.shape, '\n')

         # Se eliminan los elementos nan tanto de x como de y
         x = x[~np.isnan(y)]
         y = y[~np.isnan(y)]

         # Se cuenta el número de elementos tanto de x como de y
         print(x.shape, '\n')
         print(x.shape, '\n')

         (743,)

         (743,)

         (735,)

         (735,)
```

Figura 10:Machine Learning 01 Parte 10

Se eliminan los elementos tipo “nan” en ambos arreglos, y se procede a contar el # de elementos en ambos arreglos actualmente

```
import matplotlib.pyplot as plt

# Dibuja los puntos (x,y) con círculos de tamaño 10
plt.scatter(x, y, s=10)

# Títulos de la gráfica
plt.title("Tráfico Web en el último mes")
plt.xlabel("Tiempo")
plt.ylabel("Accesos/Hora")


plt.xticks([w*7*24 for w in range(10)],['semana %i' % w for w in range(10)])
plt.autoscale(tight=True)

# dibuja una cuadrícula punteada ligeramente opaca
plt.grid(True, linestyle='-', color='0.75')

# Muestra el gráfico
plt.show()
```

Figura 10: Machine Learning 01 Parte 10

Para finalizar con el ejemplo, se realiza la importación de la librería matplotlib y se nombra como plt, esta librería es la encargada de todo lo relacionado con las gráficas, la función .scatter nos permite definir los puntos a dibujar y el tamaño de los puntos, la función .title permite colocar el título deseado y las x.label e y.label permiten colocar tag a los ejes x e y.

plt.xticks es la encargada de definir los rangos para el ejemplo según los datos ingresados.

.grid nos permite editar la gráfica con nuevos colores, y nuevos estilos de línea, finalmente .show muestra el gráfico finalizado