Programación para *Data Science*

Unidad 4: Librerías científicas en Python

Esta Unidad se compone de 4 módulos: Matplotlib, Numpy, pandas y SciPy. A continuación, os proponemos una serie de ejercicios a realizar para cada uno de estos módulos.

NumPy - Ejercicios

Ejercicio 1

Ordenad la matriz bidimemsional `[[5,1,7], [0,7,4], [7,23,1]]` por filas utilizando como algoritmo de ordenación el Heapsort (https://es.wikipedia.org/wiki/Heapsort). **(0.5 puntos)**

In [ ]:
```
# Respuesta
```

Ejercicio 2

Definid una función que dadas dos matrices bidimensionales devuelva *True* si las dos matrices son iguales (todos sus elementos son iguales) o *False* en caso contrario. Comprobad que la matriz del ejercicio anterior es igual a la siguiente matriz: `[[12-7,1,63/9], [int(math.sin(0)),12-5,pow(2,2)], [4+3,3*7+2,pow(1,7)]]`. **(0.5 puntos)**

In [ ]:
```
# Respuesta
```

Ejercicio 3

En Python tenemos la función `bin()` que nos permite convertir un número decimal a su representación binaria. Desgraciadamente, no podemos aplicar esta función a una matriz bidimensional. Usando la función *frompyfunc* de numpy, cread una función `bin_array` que dada una matriz bidimensional devuelva otra matriz con la representación binaria de todos los elementos de la matriz original. Aplicad esta función a la matriz del Ejercicio 1 e imprimid por pantalla el resultado. **(0.5 puntos)**

In [ ]:
```
# Respuesta
```

Ejercicio 4

Cread una matriz de 8x8 que tenga un patrón 1 (blancas) / 0 (negras) como si se tratara de un tablero de ajedrez. **(0.5 puntos)**

In [ ]:
```
# Respuesta
```

Ejercicio 5

Cread dos matrices de tamaño 3x2 y 2x2 con números reales aleatorios y obtened la matriz resultado de multiplicar la primera por la segunda. **(0.5 puntos)**

In [ ]:
```
# Respuesta
```

Ejercicio 6

Calculad la norma y el determinante de la siguiente matriz: `[[1, 0], [2, -1]]`. **(0.5 puntos)**

In [ ]:
```
# Respuesta
```

Ejercicio 7

Evaluad las funciones arcoseno y arcocoseno en el intervalo [0,1] y con paso (resolución) de 0.1 y guardadlas en dos *arrays*. **(0.5 puntos)**

In [ ]:
```
# Respuesta
```

Matplotlib - Ejercicios

Ejercicio 1

Representad en un único gráfico las funciones arcoseno y arcocoseno en el intervalo [-pi/4, pi/4]. **(0.5 puntos)**

In [ ]:
```
# Respuesta
```

Ejercicio 2

Representad en dos gráficos (uno debajo del otro) las funciones f1 y f2 definidas más abajo, evaluadas en el intervalo [0, 5] y con paso (resolución) 0.02. **(0.5 puntos)**

In [1]:
```
import numpy as np

def f1(x):
    return np.exp(-x)

def f2(x):
    return np.cos(2*np.pi*x)
```

In [5]:
```
# Respuesta
```

Ejercicio 3

El fichero _salem_executions*data.csv* contiene datos sobre las acusaciones de brujería y ejecuciones llevadas a cabo en Salem durante enero de 1962 y marzo de 1963. Cargad el fichero y representad de forma gráfica el número de acusaciones versus el número de ejecuciones durante el periodo indicado. Usad una gráfica *3D* de tipo *scatter*. **(1 punto)**

In [ ]:
```
# Respuesta
```

pandas - Ejercicios

Ejercicio 1

Cargad los datos del fichero `historical_projections.csv` en un *dataframe*. Este conjunto de datos recoge resultados históricos (hasta 2015) sobre el *draft* de la NBA (https://en.wikipedia.org/wiki/NBA_draft) junto con valores probabilísticos extraídos de un modelo para predecir el éxito de los jugadores.

Mostrad el número de filas del *dataframe* y las etiquetas de los ejes. **(0.5 puntos)**

```
In [ ]: # Respuesta
```

Ejercicio 2

Agrupad los datos cargados en el ejercicio 1 según su posición (*Position*). Para cada posición, mostrad el número de jugadores que hay en el *draft* y la mediana de la probabilidad de que un jugador sea una superestrella (la probabilidad se encuentra en la columna *Superstar*). **(0.5 puntos)**

```
In [ ]: # Respuesta
```

Ejercicio 3

Mostrad los datos de los jugadores que aparecen en el *draft* de 2015 con un valor «Projected SPM» superior a 0.15 e inferior a 0.4 y que juegan o bien en la posición escolta (en inglés, *shooting guard* o *SG*) o bien en la posición ala-pívot (en inglés, *power forward* o *PF*). **(1 punto)**

```
In [ ]: # Respuesta
```

Ejercicio 4

Contad el número de jugadores que aparecen más de una vez en el *dataframe* (atendiendo al nombre del jugador), utilizando las funciones de la librería pandas. ¿Existen jugadores que aparecen múltiples veces en el *dataframe*? Comprobad si alguno de ellos aparece jugando en posiciones distintas. **(0.5 puntos)**

```
In [ ]: # Respuesta
```

Ejercicio 5

Añadid una nueva columna al *dataframe* con un valor booleano indicando si el jugador será o no una superestrella. Considerad que un jugador será una superestrella si la probabilidad de que lo sea es superior a 0.1 y tiene un valor *Projected SPM* positivo. **(1 punto)**

```
In [ ]: # Respuesta
```

SciPy - Ejercicios

Ejercicio 1: Ajustar una distribución

La variable data contiene 1000 muestras aleatorias de una distribución normal (https://en.wikipedia.org/wiki/Normal_distribution):

```
In [1]: from scipy.stats import norm

        data = norm.rvs(5, 1, size=1000)
```

Ajustad una distribución normal a las muestras de data y generad tres gráficas en una única fila (una al lado de la otra) con:

1. Un histograma con las muestras de data, usando 12 bins.
2. La función de densidad de probabilidad de la distribución ajustada.
3. El histograma y la función de densidad de probabilidad superpuestos **(1 punto)**

Pista: Podéis utilizar la función fit de scipy.stats.norm (https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.norm.html).

```
In [ ]: # Respuesta
```