Universidad de SanAndrés

Extracción y Representación de datos del Mundial

I110 - Introducción al Pensamiento Computacional

Trabajo Práctico 3

Fecha de entrega: viernes 16 de junio, 23:59 hs

Entregas tarde: se restarán 2 puntos por cadía día de entrega tarde

1. Objetivo

El objetivo de este trabajo es utilizar datos recolectados del mundial de fútbol 2022 y:

- Leer un archivo desde Python y extraer la información contenida dentro del mismo.
- Procesar la información, separar valores de texto, numéricos y de otros tipos de dato.
- Graficar los resultados para una fácil interpretación visual.
- Construir estructuras de datos que luego puedan ser almacenadas en nuevos archivos.

2. Archivos

Tenemos a disposición dos archivos que condensan un enorme conjunto de información recolectada durante el mundial:

- **"data.csv"**: Es un archivo con la información de cada partido del mundial, donde cada linea representa un partido disputado durante el mundial, y esta ordenado de primero a último (el primer partido es el que figura en la primera posición, el último partido es el que figura en la última línea). Cada columna (campo) representa una información particular de cada uno de los partidos, ordenados de la siguiente manera:

  ```
   match, dayofweek, match_time, home_team, away_team, home_xg, away_xg,
  score, attendance, venue, referee, home_formation, away_formation,
  home_captain, away_captain, home_manager, away_manager, home_possession,
  away_possession, home_completed_passes, home_attempted_pases,
  away_completed_passes, away_attempted_pases, home_sot, away_sot,
  home_total_shots, away_total_shots, home_saves, away_saves, home_fouls,
  away_fouls, home_corners, away_corners, home_crosses, away_crosses,
  home_touches, away_touches, home_tackles, away_tackles,
  home_interceptions, away_interceptions, home_aerials_won,
  away_aerials_won, home_clearances, away_clearances, home_offsides,
  away_offsides, home_gks, away_gks, home_throw_ins, away_throw_ins,
  home_long_balls, away_long_balls
  ```

  De ese conjunto de información, para este trabajo no serán de interés absolutamente todos los campos, sino particularmente los siguientes:

  - **match**: Número de partido, en el orden en que fue jugado.
  - **home_team**: Nombre del equipo que juega de "home" (local).
  - **away_team**: Equipo que juega de "away" (visitante).
  - **score**: Goles finales del partido. El formato es "[goles home]-[goles away]". En caso de haber penales el fomrato es "([penales home]) [goles home]-[goles away] ([penales away])"

- **"group_stats.csv"**: Es un archivo con la información de la fase de grupos del mundial. Los campos del archivo son:

  ```
   number, group, rank, team, matches_played, wins, draws, losses,
  goals_scored, goals_against, goal_difference, points,
  expected_goal_scored, exp_goal_conceded, exp_goal_difference,
  exp_goal_difference_per_90
  ```

  Donde nuevamente interesa el siguiente subconjunto:

  - **group**: Nº de grupo
  - **team**: País

3. Desarrollo del Trabajo Práctico

Para completar el trabajo práctico se deben completar los siguientes incisos. Antes de comenzar lea completamente la consinga y los ejercicios opcionales! A medida que vayan creando las funciones las podrán usar en los siguientes incisos.

3.1. Interpretador de Goles

Como se mencionó previamente el formato en el que se guardan los goles de cada partido si no hay penales es:

```
[goles home]-[goles away]
```

En caso de haber penales el formato es:

```
([penales home]) [goles home]-[goles away] ([penales away])
```

Haga una función que reciba un string que contenga la información de los goles de un partido (por ejemplo un empate de 1 a 1, con resultado de penales de 1 a 3, sería el string **"(1) 1-1 (3)"**) y devuelva 4 valores numéricos del tipo int:

- goles home: cantidad de goles del equipo "home"
- goles away: cantidad de goles del equipo "away"
- penales home: cantidad de penales del equipo "home". En caso de no haber penales debe ser 0
- penales away: cantidad de penales del equipo "away". En caso de no haber penales debe ser 0

3.2. Goles, Puntaje y Ranking

Haga una función que reciba el nombre de un archivo (Puede usar "datos.csv" como prueba, **pero al abrir el archivo deberá agregar encoding='utf8' para poder leer texto con tildes**) y devuelva un diccionario. En este diccionario, cada clave será el nombre de un país, y el valor asociado a cada una será otro diccionario. Cada uno de estos diccionarios debe tener la información de los goles totales convertidos por el país a lo largo de **todo** el mundial, puntos obtenidos en la fase de grupos, y ranking final. La estructura quedaría de la siguiente manera:

```
teams = {'paísx': { 'goals': x, 'points': [p0, p1, p2, p3], 'rank':x },
         'paísy': { 'goals': y, 'points': [p0, p1, p2, p3], 'rank':0 },
         'paísz': { 'goals': z, 'points': [p0, p1, p2, p3], 'rank':0 },
        ...
        }
```

Donde p0 es el puntaje al principio de la fase de grupos (todos 0), p1 el puntaje después del primer partido, p2 del segundo y p3 del tercero. Para calcular estos puntajes deberá utilizar la información de los partidos del archivo, pero tenga en cuenta que el archivo contiene información de los partidos que vienen luego de la fase de grupos. Recuerde que:

- Partido ganado = +3 puntos
- Partido perdido = +0 puntos
- Partido empatado = +1 punto

El ranking final será simplemente un valor numérico que indica la posición final del país en el mundial, la cual vale 1, 2, 3 o 4 para los equipos que quedaron en los primeros cuatro puestos, y el valor 0 en cualquier otro caso (fuera del podio, no hay un orden de posiciones).

3.3. Agregar datos

Haga una función que reciba el nombre de un archivo que contenga información de los grupos (use "group_stats.csv" para probar su código, **pero al abrir el archivo deberá agregar encoding='utf8' para poder leer texto con tildes**) y el diccionario que creó anteriormente, y le agregue a cada país el grupo al cual pertenece agregando otra clave y valor al diccionario de cada país. Quedaría entonces:

```
teams={'paísx':{'goals': x, 'points':[p0, p1, p2, p3], 'rank':x, 'group':a},
       'paísy':{'goals': y, 'points':[p0, p1, p2, p3], 'rank':0, 'group':b},
       'paísz':{'goals': z, 'points':[p0, p1, p2, p3], 'rank':0, 'group':c},
        ...
      }
```

Donde **a**, **b** y **c** representan los números de grupo a los que pertenece cada país.

3.4. Ranking de Goles

Haga una función que tome un diccionario de diccionarios y una clave deseada, y genere dos listas. La primer lista está compuesta por las claves del diccionario, (en el caso del diccionario que creamos hasta ahora serían los nombres de los países). La segunda lista contiene el valor que guarda cada sub-diccionario para la clave especificada (para el caso del diccionario anterior podría ser "goals" por ejemplo). Ambas listas deben estar ordenadas según los valores de la segunda lista. Por ejemplo, si la llave especificada es "goals" y mi diccionario es:

```
teams={'España':{'goals': 10, 'points':[0, 3, 4, 5], 'rank':0, 'group':a},
       'Alemania':{'goals': 12, 'points':[0, 1, 4, 4], 'rank':0, 'group':b},
       'Inglaterra':{'goals': 8, 'points':[0, 0, 0, 3], 'rank':0, 'group':c}}
```

Las dos listas generadas deberían ser:

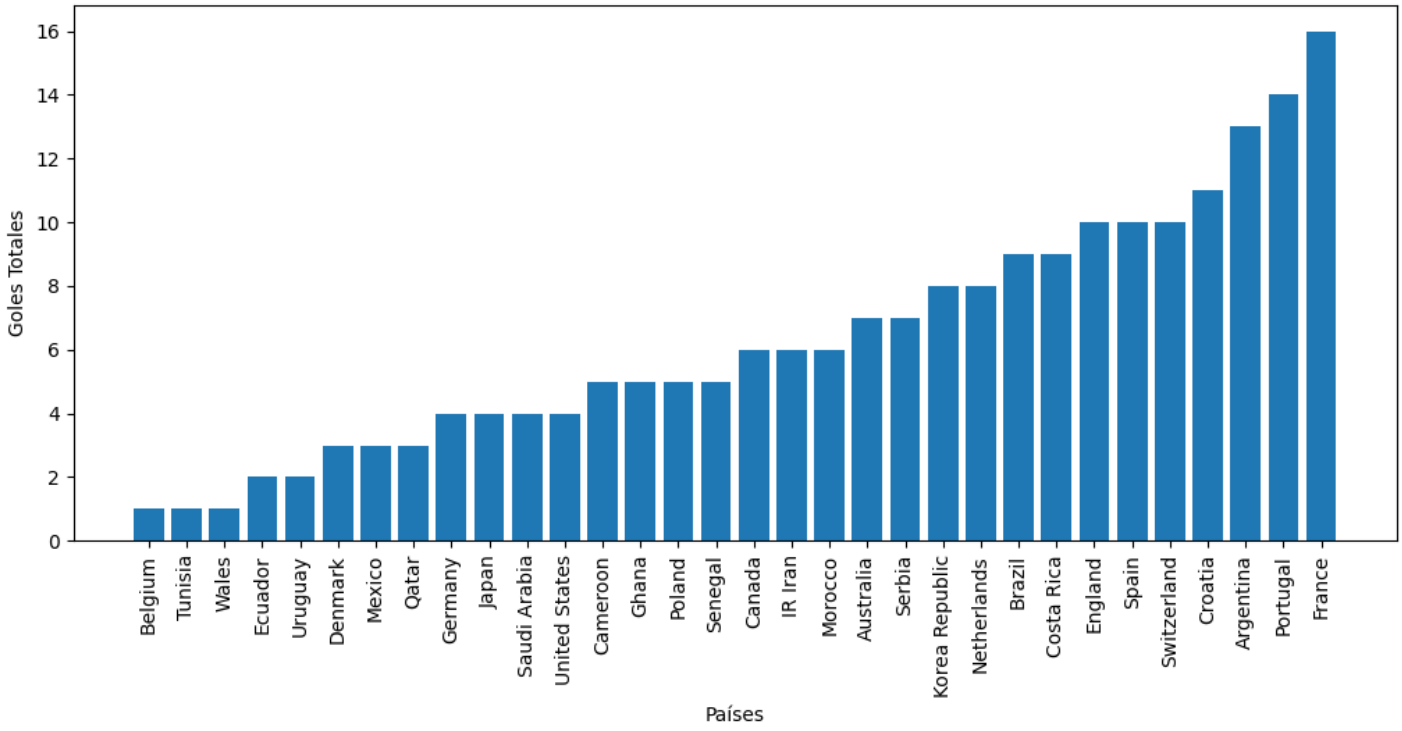
```
A = ['Inglaterra', 'España', 'Alemania']
B = [8, 10, 12]
```

Nótese que se cambió el orden de los países para que queden ordenados según la cantidad de goles.

> HINT: Revise que pasa si hace "**sorted(zip(listaB, listaA))**". Por ejemplo, si **listaA** es **['a', 'b', 'c']** y **listaB** es **[3, 2, 1]**, que vale la variable **resultado** luego de hacer **resultado = sorted(zip(listaB, listaA))**?

3.5. Gráfico de Goles

Haga una función que reciba el diccionario y grafique con barras los goles totales de cada equipo en orden ascendente. Para que se vean bien los nombres de paises deben estar rotados 90 grados. El gráfico debe quedar similar al siguiente:

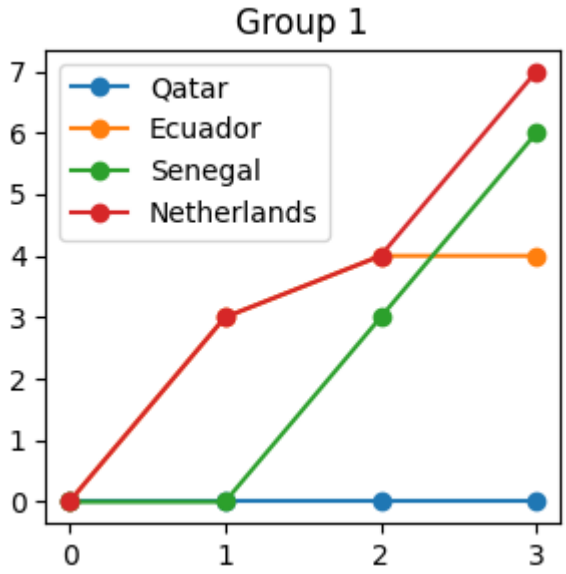


Su gráfico debe tener los nombres rotados, el nombre del eje x, y el nombre del eje y.

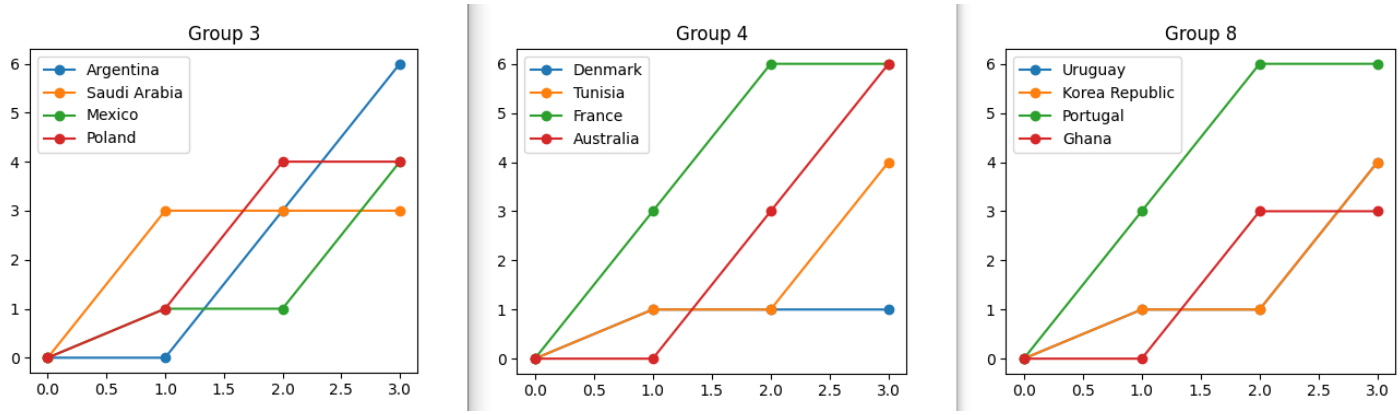
> Tip: Use la función creada en el punto anterior para extraer los datos

3.6. Gráficos de Grupos

Haga una función que reciba el diccionario y una lista de grupos y grafique (Una figura por grupo) los puntajes cumulativos de cada equipo en cada grupo indicado. Si la lista de grupos fuera [1] las figuras deberían ser:



Note que el gráfico tiene una leyenda para saber que equipo es que línea, y que las lineas tienen puntos para cada etapa. Si la lista de grupos fuera [3, 4, 8] las figuras deberían ser:



> Nota: Para este ejercicio le puede ser útil primero crear una función que reciba un diccionario y un grupo y que genere una figura para ese grupo en particular

3.7. Guardado de datos

Haga una función que guarde la información almacenados en el diccionario del punto 3.3 en un archivo con fomrato **.csv**, siendo los campos (columnas) del archivo:

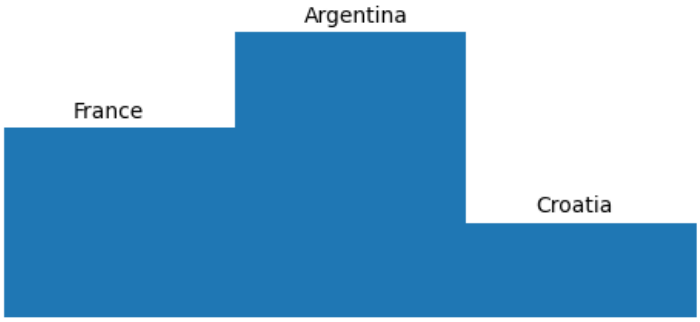
- Team: Nombre del país
- Goals: Goles totales
- Group: Grupo al cual pertenece el equipo
- Score1: Puntaje después del primer partido de la fase de grupos
- Score2: Puntaje después del segundo partido de la fase de grupos
- Score3: Puntaje después del tercer partido de la fase de grupos
- Rank: Ranking del equipo en el mundial (0, 1, 2, 3 o 4)

4. Extensiones [OPCIONAL]

Si usted desea, puede agregar funcionalidades más allá de las que se piden en el punto 3., siempre y cuando no interfieran con las funcionalidads básicas. Si estas funcionalidades "extra" funcionan correctamente y no interfieren con el flujo básico del juego, el docente podrá a su discreción darle hasta 2 puntos adicionales, que levantan la nota en caso que no sea la máxima (si la nota base era un 8 pero se cumplen con todos los puntos opcionales se puede subir a un 10).

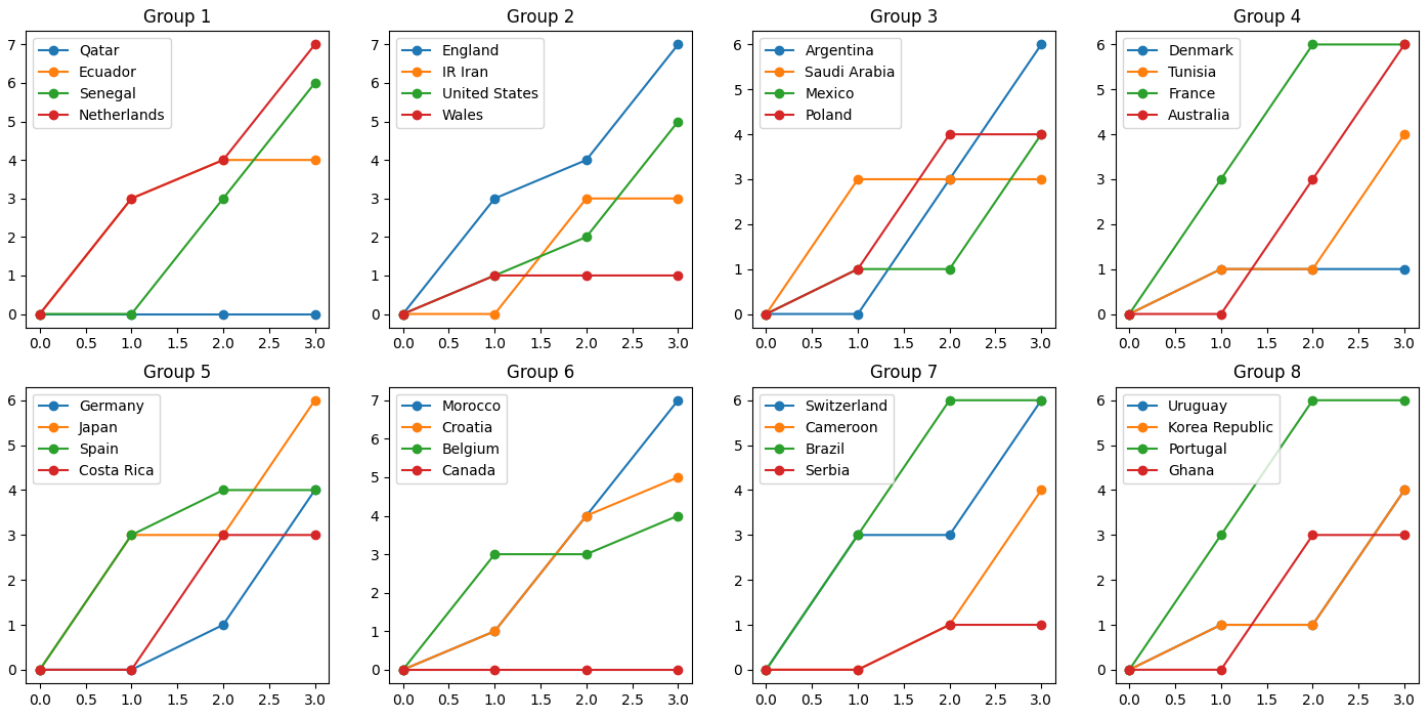
A continuación se muestran ideas de "features" adicionales que usted podría agregar.

1. [★] Haga una función que tome el diccionario de equipos y grafique un podio. Para que valga el punto el podio debe tener los nombres de los paises y no tener ejes de coordenadas como se ve a seguir:



2. [★ ★] Para su función del problema 3.4 agregue una funcionalidad para que también se pueda ordenar los paises según su puntaje en la etapa de grupos. Se deberá poder elegir en base a que ronda de partidos ordenar las listas (Ej: Ordenar según los puntajes luego de la segunda ronda de partidos)

3. [★ ★ ★] Para el ejercicio 3.6 donde se grafican los puntajes en la fase del grupo, modifique su función para que si no se recibe una lista de grupos automáticamente se haga **un** (todo en uno) gráfico con todos los grupos. Debería quedar de la siguiente manera:



5. Modalidad de Trabajo y Formato de entrega

- El TP se podrá realizar en grupos de hasta 2 personas.
- Se debe realizar una entrega digital a través del Campus Virtual de la materia, compuesta por el archivo con el código fuente en .py.
- **El nombre del archivo deberá seguir el formato** *tp3_apellido1_apellido2.py*
- El código entregado deberá estar debidamente comentado de manera que el docente que corrige el trabajo entienda lo que se está haciendo y por qué decidieron hacerlo de tal o cual manera. Además, todas las funciones deben contener su docstring, indicando parámetros de entrada y salida, tipos de datos, y breve descripción de la función. Abajo se muestra un ejemplo de un código con comentarios adecuados. El programa en cuestión debe imprimir en pantalla los números múltiplos de 5 hasta que el usuario decida que no se impriman más. Observe cómo está comentado para que se comprenda el flujo. Ustedes deberán comentar su código de manera similar para que el corrector entienda lo que querían hacer.

```
def multi_5():
    """
    La función multi_5 imprime numeros múltiplos de 5 hasta que el usuario
    decida no imprimir más. Para que eso suceda deberá ingresar un valor
    distinto de 'si'. Una vez que se deja de imprimir el programa devuelve
    el último valor al que se llegó
    Entrada:
        - None
    Salida:
        - final [int]: Último numero al que se llegó
    """

    # creamos la variable "numero" que guardará el número a imprimr en cada
    # ciclo. Como vamos a sumarle 5 cada vez que se corra el ciclo la variable
    # comienza en 0 para que en el primer ciclo se le sume 5 y se imprima 5
    numero = 0

    # Creamos la variable "seguir" que indicará si se quiere seguir
    # imprimiendo números Para que el ciclo se haga por lo menos una vez se
    # comienza con 'si'
    seguir = 'si'

    # El ciclo while seguirá siempre y cuando "seguir" sea 'si'
    while seguir == 'si':
        numero += 5 # se le suma 5 para ir por múltiplos de 5
        print(numero)
        seguir = input('¿Quiere seguir?    ') # Se le pregunta al usuario si
        # quiere seguir y se guarda la respuesta en la variable "seguir"
        # Si el usuario ingresa 'si' se repetirá el ciclo

    # Una vez que el usuario ingrese algo que no sea 'si' se rompe el ciclo y
    # se termina el código
    print('Adiós')
    return numero
```

```
# Utilizamos la función
valor = multi_5()
```