prog_datasci_8_vis_entrega

May 18, 2022

1 Fundamentos de Programación

1.1 PEC 8 - Visualización de datos en python

En este Notebook se encontraréis el conjunto de actividades evaluables como PEC de la asignatura. Veréis que cada una de ellas tiene asociada una puntuación, que indica el peso que tiene la actividad sobre la nota final de la PEC. Adicionalmente, hay un ejercicio opcional, que no tiene puntuación dentro de la PEC, pero que se valora al final del semestre de cara a conceder las matrículas de honor y redondear las notas finales. Podréis sacar la máxima nota de la PAC sin necesidad de hacer este ejercicio. El objetivo de este ejercicio es que sirva como pequeño reto para los estudiantes que quieran profundizar en el contenido de la asignatura.

Veréis que todas las actividades de la PEC tienen una etiqueta, que indica los recursos necesarios para llevarla a cabo. Hay tres posibles etiquetas:

- NM **Sólo materiales**: las herramientas necesarias para realizar la actividad se pueden encontrar en los materiales de la asignatura.
- EG **Consulta externa guiada**: la actividad puede requerir hacer uso de herramientas que no se encuentran en los materiales de la asignatura, pero el enunciado contiene indicaciones de dónde o cómo encontrar la información adicional necesaria para resolver la actividad.
- EI **Consulta externa independiente**: la actividad puede requerir hacer uso de herramientas que no se encuentran en los materiales de la asignatura, y el enunciado puede no incluir la descripción de dónde o cómo encontrar esta información adicional. Será necesario que el estudiante busque esta información utilizando los recursos que se han explicado en la asignatura.

Es importante notar que estas etiquetas no indican el nivel de dificultad del ejercicio, sino únicamente la necesidad de consulta de documentación externa para su resolución. Además, recordad que las **etiquetas son informativas**, pero podréis consultar referencias externas en cualquier momento (aunque no se indique explícitamente) o puede ser que podáis hacer una actividad sin consultar ningún tipo de documentación. Por ejemplo, para resolver una actividad que sólo requiera los materiales de la asignatura, puedéis consultar referencias externas si queréis, ya sea tanto para ayudaros en la resolución como para ampliar el conocimiento!

En cuanto a la consulta de documentación externa en la resolución de los ejercicios, recordad **citar siempre la bibliografía utilizada** para resolver cada actividad.

1.1.1 Ejercicio 1 (seaborn)

En el siguiente dataset en `data/starclass.csv`, encontraréis datos de las estrellas a nuestro alrededor, con información de:

- Temperatura absoluta (en K)
- Luminosidad relativa (L/Lo)
- Radio relativo (R/Ro)
- Magnitud absoluta (Mv)
- Color (white,Red,Blue,Yellow,yellow-orange etc)
- Clase espectral (O,B,A,F,G,K,,M)
- Tipo de estrella
  - Enana roja (Red Dwarf)
  - Enana marrón (Brown Dwarf)
  - Enana blanca (White Dwarf)
  - Secuencia principal (Main Sequence)
  - Supergigantes (SuperGiants)
  - Hipergigantes (HyperGiants)

Nota: * Lo = 3.828 x 10^26 Watts (Avg Luminosity of Sun) * Ro = 6.9551 x 10^8 m (Avg Radius of Sun)

a) Carga el archivo de datos en un objeto pandas.

Muestra:

- El número de estrellas en el conjunto de datos
- Los nombres de las variables.
- Las últimas 10 entradas,

NM **(0.5 puntos)**

[1]:
```
# Respuesta
```

b) Muestra:

- Un histograma de la luminosidad (con `displot`)
- Un histograma de los radios. (con `displot`)
- Un diagrama de puntos con las magnitudes absolutas en el eje y y la Temperatura en el eje x. (con `jointplot`)

NM **(0.5 puntos)**

[8]:
```
# Respuesta
```

c) Parece que existe una clara correlación entre las variables de la base de datos. Utiliza un mapa de calor para visualizar las correlaciones entre los atributos numéricos de este último dataframe de forma que:

- Visualiza el mapa de color de forma que aparezcan los valores de correlación en la figura, con dos decimales
- Usa un mapa de color en el que los valores de correlación más elevada sean colores más oscuros.

- Haz que en el mapa de color sólo aparezca la semidiagonal inferior (sin mostrar la diagonal). Puedes consultar el argumento `mask` de la documentación de seaborn (https://seaborn.pydata.org/generated/seaborn.heatmap.html)
- ¿Qué par de variables están más correlacionadas?

EG **(1 punto)**

[10]:
```
# Respuesta
```

Ejercicio 2

A continuación, veremos que las estrellas siguen un determinado patrón en el espacio celeste, específicamente llamado Diagrama Hertzsprung-Russell o simplemente Diagrama-HR. Podemos clasificar a las estrellas dibujando sus características a partir de este gráfico.

a) Realiza los siguientes pasos:

- Muestra los valores de la variable de `Star color`.
- Corrige los duplicados y errores de los valores de forma que la codificación sea única y uniforme.
- Muestra los valores únicos de color después de la corrección.

EG **(0.5 puntos)**

[28]:
```
# Respuesta
```

b) Codifica el tipo estelar (`Star type`) de forma que aparezca el nombre, y no el código numérico, según esta tabla:

0: Enana roja (Red Dwarf) 1: Enana marrón (Brown Dwarf) 2: Enana blanca (White Dwarf) 3: Secuencia principal (Main Sequence) 4: Supergigantes (SuperGiants) 5: Hipergigantes (HyperGiants)

NM **(0.5 puntos)**

[45]:
```
# Respuesta
```

c) Respresenta en un `boxplot` de seaborn las siguientes relaciones:

- La Magnitud absoluta vs el tipo estelar.
- La Temperatura vs el tipo estelar.

De forma adicional al boxplot, muestra en ambas figuras la ubicación de los datos en color negro mediante un stripplot.

EG **(0.5 puntos)**

[49]:
```
# Respuesta
```

d) Construye el Diagrama Hertzsprung-Russell en un `scatterplot` de seaborn con la siguiente configuración:

- x: Temperaturas
- y: El logaritmo de la luminosidad

- Color: La clase espectral
- Tamaño de los puntos: El Radio
- El estilo del punto: el tipo estelar

Nota: Crea una figura de 10x7 para realizar el plote. Modifica la transparencia de los puntos (alfa) a 0.5.

NM **(1 punto)**

[72]:
```
# Respuesta
```

1.1.2 Ejercicio 3 (NETWORKX)

a) Crea un grafo no dirigido de 70 nodos de forma que:

- Los nombres de los nodos sean `N1`, $\cdots$, `N70`
- Dos nodos estén conectados si un entero es divisor del otro. Ejemplos:
  - `N48` conectará con `N24`
  - `N5` no conectará con `N7`
  - `N10` conectará con `N5`
  - `N3` no conectará con `N2`
  - `N70` conectará con `N35`

Visualiza el grafo resultante de forma que aparezcan los nombres de los nodos.

EI **(1 punto)**

[14]:
```
# Respuesta
```

b) Borra los nodos N1, N2, N3, N4 y N5. Muestra las aristas del nodo 'N27'

NM **(0.5 punto)**

[238]:
```
# Resposta
```

c) Calcula el tamaño de todos los componentes conexos del grafo. Encuentra programáticamente el mayor componente conexo del grafo, es decir, el sugrafo más grande.

Representa sólo el subgrafo mayor de modo que los nodos tengan un color proporcional a la centralidad de cada nodo y que aparezca el nombre del nodo.

Nota: Usar eigenvector centrality como aquí

EG **(1.5 puntos)**

[237]:
```
# Respuesta
```

1.1.3 Ejercicio 4 (GEOPLOTLIB)

A lo largo de la historia de los Juegos Olímpicos éstos se han realizado en múltiples ubicaciones en todo el mundo. En este ejercicio intentaremos visualizar en un mapa esta distribución.

a) Para realizar una visualización básica de la localización geográfica de las sedes de los Juegos Olímpicos, pedimos:

- Carga los datos en un dataframe de pandas conteniendo los datos con la localización de las principales ciudades del planeta del archivo `worldcities.csv`
- En un segundo dataframe, carga el archivo con los datos de los Juegos Olímpicos del archivo `olimpics.csv`.
- Crea un tercer dataframe, con las siguientes columnas para cada uno de los eventos olímpicos [year,city,country,population,continent,lat,lon]

NM **(0.5 puntos)**

[ ]:
```
# Respuesta
```

b) Visualiza un mapa del mundo con las posiciones en las que se han celebrado los Juegos Olímpicos, utilizando el dataframe generado anteriormente y la librería geoplotlib.

NM **(1 punto)**

[1]:
```
# Respuesta
```

c) Existen algunas ciudades que han repetido celebraciones. En éste ejercicio deberemos:

- Cambiar el color de los puntos que hemos utilizado para ubicar cada ciudad en el apartado anterior para visualizar utilizando distintos colores las veces que se han celebrado los juegos en la misma ciudad.
- Añadir una etiqueta, con el número de repeticiones junto al punto.

EI **opcional**

[2]:
```
# Respuesta
```

1.1.4 Ejercicio 5 (MATPLOTLIB)

A partir de una web de referencia de datos de sanidad (en https://github.com/datadista/datasets/tree/master/COVID%2019) hemos descargado datos históricos de casos de infección del virus sars-cov-2 y su severidad, que tiene en el fichero el archivo `provincias_covid19_datos_sanidad_nueva_serie.csv`, donde verá los casos, hospitalizaciones, ingresos en UCI y defunciones, por provincia y fecha.

En el último ejercicio representaremos la evolución a lo largo del tiempo de las variables del conjunto de datos usando `matplotlib`.

- Importa los datos como un objeto pandas.
- Construye una gráfica con 4 subplots diferentes (2x2, es decir, dos filas y dos columnas) mostrando la evolución temporal de cada índice por los casos de Madrid y Barcelona, indicando las fechas en el eje `x`:

```
|--------------------------------|
| Casos              | Defunciones|
|--------------------|------------|
```

```
| Hospitalizaciones | UCI        |
|-------------------------------|
```

Recuerda etiquetar las figuras, ejes e incluir leyendas. Rota 45 grados las etiquetas de los ejes temporales si crees que ayuda a su lectura.

Nota: Puedes utilizar el módulo `datetime` para procesar la columna de fechas.

NM **(1 punto)**

[ ]:
```
# Respuesta
```

Ejercicio Opcional

Dada la serie del barco ardiente fractal:

$$z_{n+1} = (|\operatorname{Re}(z_n)| + i|\operatorname{Im}(z_n)|)^2 + c$$

y dado un elemento inicial de la serie $z_0 = 0 + i0$

1. Escribe una función $f(c, A, N_{max})$, que devuelva el número iteraciones N de la serie anterior necesarias para que el módulo $|z| > A$, siendo $A \in \mathbb{R}$.
2. Calcula y representa una imagen 2D representando N sobre el dominio de c comprendido por $Re(c) \subset [-2, 1]$, $Im(c) \subset [-2, 1]$ con los parámetros ($A = 2$, $n_{max} = 100$, $z_0 = i0$, Resolución de la imagen = (800,1200) pixels).

Nota: Puede ser útil la función `meshgrid` de `numpy`.

EI

[357]:
```
# Resposta
```