CAT-PAC4-enun

December 28, 2024

```
<div style="float: left; width: 50%;">
   <img src="https://www.uoc.edu/portal/system/modules/edu.uoc.presentations/resources/img/bra
</div>

<p style="margin: 0; padding-top: 22px; text-align:right;">22.403 · Programació per a la ciènc
<p style="margin: 0; text-align:right;">Grau en Ciència de Dades Aplicada</p>
<p style="margin: 0; text-align:right; padding-button: 100px;">Estudis d'Informàtica, Multimèd
```

1 Programació per a la ciència de dades - PAC4

En aquest Notebook trobareu l'exercici que suposa la quarta i darrera activitat d'avaluació continuada (PAC) de l'assignatura. Aquesta PAC intenta presentar-vos un petit projecte en el qual heu de resoldre diferents exercicis, que engloba molts dels conceptes coberts durant l'assignatura.

L'objectiu d'aquest exercici serà desenvolupar un paquet de Python fora de l'entorn de Notebooks, que ens permeti resoldre el problema donat. Treballareu amb arxius Python plans `.py`. El projecte haurà d'incloure el corresponent codi organitzat lògicament (separat per mòduls, organitzats per funcionalitat,...), la documentació del codi (*docstrings*) i tests. A més, haureu d'incloure els corresponents arxius de documentació d'alt nivell (`README`) així com els arxius de llicència i dependències (`requirements.txt`) comentats a la teoria.

Fer un `setup.py` és opcional, però si es fa es valorarà positivament de cara a la nota de la pràctica i del curs.

2 Enunciat: Orbea Monegros 2024

L'**Orbea Monegros** és una prova de ciclisme de muntanya (BTT) no competitiva que es realitza a Sariñena (Osca). La classificació es pot consultar a l'enllaç, amb els noms reals dels ciclistes, però nosaltres hem preferit anonimitzar les dades dels ciclistes, que proporcionem en el dataset que trobaràs a la carpeta *data/*.

- https://www.orbea.com/es-es/eventos/monegros-2024
- https://sportmaniacs.com/es/races/orbea-monegros-2024/662d72e2-07c8-4360-be3d-4072ac1f171c/results#rankings

3 Projecte Python, funcionalitat

Per fer el lliurament més fàcil i homogeni us demanem que organitzeu el codi de tal manera que **des del fitxer principal retorneu totes les respostes que se us demana a la PAC** fent ús de funcions que haureu de definir en mòduls. Per això, en cada exercici us indicarem el format que ha de tenir cada resposta, de manera que executant `main.py` es vagi responent a tota la PAC. Per defecte, `main.py` ha d'executar totes les funcions de la PAC mostrant com funcionen però també ha de permetre executar-les una per una si es desitja (amb el benentès que per executar la segona s'ha d'haver executat prèviament la primera, etc). Ho heu de documentar tot molt bé en el *README* de manera que el professor pugui executar el codi sense problemes i sense dubtes. Us recordem que al *README* també heu d'indicar com executar els tests i comprovar-ne la cobertura.

3.1 Ex1 (1p). Importació del dataset i EDA

- Importació del dataset a dataframe.
- Mostrar els 5 primers valors
- Quants ciclistes van participar a la prova?
- Quines columnes té el dataframe?

3.2 Ex2 (1,5p). Anonimitzar els ciclistes. Netejar el dataset

Tot i que les dades que et proporcionem ja estan anonimitzades, anem a suposar que són les dades reals, i que les volem anonimitzar. Pots utilitzar la llibreria *Faker* per generar noms i cognoms (en anglès).

Defineix la funció *name_surname(df)*, que li passarem com a argument el dataframe, i ens retornarà el nou dataframe amb els noms canviats (columna *biker*).

Executa la funció, i mostra els 5 primers valors del dataframe.

Els ciclistes que tenen un temps *00:00:00* significa que no van participar a la prova (estaven inscrits però no van fer la cursa). Elimina'ls del dataset.

Quants ciclistes tenim ara en el dataframe? Mostra els 5 primers. Recupera les dades del ciclista amb dorsal=1000.

3.3 Ex3 (1,5p). Agrupament dels minuts. Histograma

Volem fer un anàlisi estadístic del temps que han trigat els ciclistes a completar la prova (108 Km). Per tal de què surti un histograma significatiu es vol agrupar els temps en franges de 20 min. De manera que: * Si un ciclista ha trigat 6h 19min 40seg se li assignarà 06:00. * Si un ciclista ha trigat 6h 29min 40seg se li assignarà 06:20. * Si un ciclista ha trigat 6h 59min 40seg se li assignarà 06:40.

Defineix la funció *minutes_002040()* on passem un valor de temps en format hh:mi:ss i retornarà un valor en format hh:mi, on el valor dels minuts només pot agafar els valors 00, 20 o 40.

Amb aquesta funció crea una nova columna *time_grouped* al dataframe. Mostra els 15 primers valors del dataframe. Aquesta nova columna ens servirà per fer l'histograma.

Ja pots fer un agrupament (*groupby*) per la columna *time_grouped*, de manera que obtindràs un nou dataframe amb els valors de la columna *time_grouped* i el número de ciclistes que entren dins de cada franja. Mostra els valors d'aquest nou dataframe.

Amb aquesta informació ja pots generar l'histograma, que hauràs de guardar a *img/histograma.png*.

3.4 Ex4 (1,5p). Clubs ciclistes

Ens fixem ara en els clubs ciclistes dels corredors (si no pertanyen a cap club estan en la categoria *Independiente*). Però tenim un problema: en el formulari d'inscripció el camp *club* és lliure per posar qualsevol cosa. I ens trobem valors com *Club Ciclista Huesca*, CC Huesca, C.C. Huesca, C.C.Huesca o Huesca, i encara podria haver d'altres possibilitats.

Volem netejar (bastant, sense pretendre fer-ho perfecte) el nom dels clubs, i seguirem aquestes regles: * convertim el nom del club a majúscules * reemplacem per no res els següents valors: 'PEÑA CICLISTA', 'PENYA CICLISTA', 'AGRUPACIÓN CICLISTA', 'AGRUPACION CICLISTA', 'AGRUPACIÓ CICLISTA', 'AGRUPACIO CICLISTA', 'CLUB CICLISTA', 'CLUB' * reemplacem per no res els següents valors quan estan a l'inici (expressió regular): 'C.C.', 'C.C', 'CC', 'C.D.', 'C.D', 'CD', 'A.C.', 'A.C', 'AC', 'A.D.', 'A.D', 'AD', 'A.E.', 'A.E', 'AE', 'E.C.', 'E.C', 'EC', 'S.C.', 'S.C', 'SC', 'S.D.', 'S.D', 'SD'. Fixa't bé que hem utilitzat espais en blanc després de la cadena. * reemplacem per no res els següents valors quan estan al final: ' T.T.',' T.T',' TT',' T.E.',' T.E',' TE',' C.C.',' C.C',' CC',' C.D.',' C.D',' CD',' A.D.',' A.D',' AD',' A.C.',' A.C',' AC' * Finalment, elimina possibles espais en blanc al principi o al final de la cadena.

Defineix la funció *clean_club()*, on li passem un club i ens retorna el club amb els valors netejats. Per exemple, 'C.C. Huesca' retorna 'Huesca'.

Crea una nova columna *club_clean* al dataframe, amb els valors nets. Mostra els 15 primers valors.

Ara ja podem crear un nou dataframe amb les dades agrupades a partir de la nova columna, on es mostrarà el número de ciclistes participants de cada club.

Ordena el dataframe pel número de clubs amb més participants. El club 'INDEPENDIENTE' és el que en té més, com és lògic, però després vénen els clubs UCSC, SARIÑENA, OSCENSE, etc.

4 Ex5 (1p). Unió Ciclista Sant Cugat (UCSC)

A partir del dataframe obtingut en el problema anterior, contesta les següents preguntes:

- Quins són els ciclistes de la UCSC (Unió Ciclista Sant Cugat)?
- Quin ciclista de la UCSC ha fet millor temps?
- En quina posició sobre el total ha quedat aquest ciclista, i quin percentatge sobre el total representa?

5 Criteris de correcció

Aquesta PAC s'avaluarà seguint els criteris següents:

- **Funcionalitat** (6.5 punts): Es valorarà que el codi implementi tot el que es demana.
  - Exercici 1 (1 punt)
  - Exercici 2 (1,5 punts)
  - Exercici 3 (1,5 punts)
  - Exercici 4 (1,5 punts)
  - Exercici 5 (1 punt)

- **Documentació** (0,5 punts): Totes les funcions dels exercicis d'aquesta PAC hauran d'estar degudament documentades utilitzant docstrings (en el format que preferiu).
- **Modularitat** (0,5 punts): Es valorarà la modularitat del codi (tant l'organització del codi en mòduls com la creació de funcions). S'espera que, a part del script main.py, hi hagi també un fitxer per cada exercici.
- **Estil** (0,5 punts): El codi ha de seguir la guia d'estil de Python (PEP8), exceptuant els casos on fer-ho compliqui la llegibilitat del codi. Per demostrar-ho, els alumnes executaran pylint (documentar al README).
- **Tests** (1,5 punts): El codi ha de contenir una o diverses suites de tests que permetin comprovar que el codi funciona correctament, amb un mínim del 50% de cobertura (documentar al README).
- **Requeriments** (0,25 punts): Heu d'incloure un fitxer de *requirements.txt* que contingui la llista de llibreries necessàries per executar el codi.
- **README** i **LICENSE** (0,25 punts): Heu d'afegir també un fitxer README, que presenti el projecte i expliqui com executar-lo, així com la inclusió de la llicència sota la qual es distribueix el codi (podeu triar la que vulgueu). Concretament, es demana que s'expliqui com s'executa el projecte des del principi fins al final o fins a un cert punt; com s'executen els tests i la cobertura; com es comprova l'estil amb pylint.

5.0.1 Important

Nota 1: De la mateixa manera que en les PACs anteriors, els criteris transversals es valoraran de manera proporcional a la part de funcionalitat implementada.

Per exemple, si el codi només implementa la meitat de la PAC i la documentació està perfecta, la puntuació corresponent a la documentació serà de 0,25.

Nota 2: És imprescindible que el paquet que lliuris s'executi correctament en l'entorn que hagis fet servir (Anaconda, etc.) i que el fitxer *REAMDE* expliqui clarament com executar el codi per generar els resultars que es demanen. A més, en el *README* s'ha d'explicar també com s'executaran els tests i com es comprovarà la seva cobertura.

Nota 3: Entregueu el paquet como un únic arxiu .zip que contingui només el codi en el Registre d'Avaluació Contínua. **El codi Python estarà escrit amb fitxers plans de Python.**