Projeto 2 Astrometria

André Almeida Trovello

Estudo de regiões de formação estelar (Taurus) com métodos de MAchine Learning

Query

Importando Bibliotecas

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.cm as cm
from astroquery.gaia import Gaia
from sklearn.cluster import KMeans, DBSCAN
from sklearn.metrics import silhouette_score, silhouette_samples
from astropy.table import Table
```

Criando Dataframe

```
data = pd.read_csv('galli_2019_table1.csv')
main_table = pd.DataFrame(data)
```

Tabela de Vizier

```
# Garanta que a variável 'main_table' exista.
# Se você estiver rodando o script do zero, você precisa
# carregar o arquivo CSV que você salvou do VizieR.
try:
    # Se você já tiver 'main_table' na memória (ex: num notebook), pule
    main_table
    print("Variável 'main_table' já existe na memória.")
except NameError:
    try:
        # Tente carregar do CSV
        # Substitua 'galli_2019_table1.csv' pelo nome que você usou
        main_table = Table.read('galli_2019_table1.csv')
        print("Arquivo 'galli_2019_table1.csv' carregado com sucesso.")
    except FileNotFoundError:
        print("Erro: A variável 'main_table' não foi encontrada na memória.")
        print("E o arquivo 'galli_2019_table1.csv' não foi encontrado no disco.")
        print("Por favor, rode o script do VizieR primeiro ou verifique o nome do arquivo CS
        raise
```

```
Variável 'main_table' já existe na memória.
```

Limpar os IDs

Remove “Gaia DR2” dos nomes dos objetos mantendo apenas seus IDs

```
print("Iniciando a limpeza dos IDs...")
try:
    dr2_id_strings = main_table['GaiaDR2'].tolist()

    dr2_ids_cleaned_list = []
    ids_pulados = 0

    for s in dr2_id_strings:
        try:
            id_str = s.split(' ')[-1]
            numeric_id = np.int64(id_str)
            dr2_ids_cleaned_list.append(numeric_id)
        except (AttributeError, IndexError, ValueError):
            ids_pulados += 1

    print(f"Encontrada e limpa a lista de {len(dr2_ids_cleaned_list)} IDs (DR2) válidos.")
```

```
    if ids_pulados > 0:
        print(f"({ids_pulados} linhas foram puladas por não terem um ID válido)")

except (KeyError) as e:
    print(f"Erro: A coluna 'GaiaDR2' não foi encontrada na 'main_table'.")
    raise
```

```
Iniciando a limpeza dos IDs...
Encontrada e limpa a lista de 458 IDs (DR2) válidos.
(61 linhas foram puladas por não terem um ID válido)
```

Cria a tabela para upload e a query organizada

```
upload_table = Table({'dr2_source_id_list': dr2_ids_cleaned_list})

# QUERY CORRIGIDA E OTIMIZADA:
# 1. Começa com a sua tabela (user_table)
# 2. Usa o nome correto da tabela (gaiadr3.dr2_neighbourhood)
query_dr3 = """
SELECT
    dr3.source_id, dr3.ra, dr3.dec, dr3.parallax, dr3.pmra, dr3.pmdec, dr3.ruwe
FROM
    tap_upload.my_table AS user_table
JOIN
    gaiadr3.dr2_neighbourhood AS xmatch
    ON user_table.dr2_source_id_list = xmatch.dr2_source_id
JOIN
    gaiadr3.gaia_source AS dr3
    ON xmatch.dr3_source_id = dr3.source_id
"""
```

Executa a busca

```
print("Iniciando a busca no Gaia DR3 (com query otimizada)...")

try:
    job = Gaia.launch_job_async(
        query=query_dr3,
```

```
        upload_resource=upload_table,
        upload_table_name="my_table",
        verbose=True  # Adiciona mais informações de debug
    )

    results_dr3_members = job.get_results()

    print(f"\nBusca concluída!")
    print(f"Foram encontrados dados no DR3 para {len(results_dr3_members)} das {len(dr2_ids_

    print("\n--- 5 primeiras linhas dos membros de Taurus (dados do DR3) ---")
    print(results_dr3_members.to_pandas().head())

    # Salve os resultados se desejar
    results_dr3_members.write('taurus_membros_dr3.csv', format='csv', overwrite=True)

except Exception as e:
    print(f"\nOcorreu um erro durante a busca no Gaia:")
    print(e)
```

```
Iniciando a busca no Gaia DR3 (com query otimizada)...
Launched query: '
SELECT
    dr3.source_id, dr3.ra, dr3.dec, dr3.parallax, dr3.pmra, dr3.pmdec, dr3.ruwe
FROM
    tap_upload.my_table AS user_table
JOIN
    gaiadr3.dr2_neighbourhood AS xmatch
    ON user_table.dr2_source_id_list = xmatch.dr2_source_id
JOIN
    gaiadr3.gaia_source AS dr3
    ON xmatch.dr3_source_id = dr3.source_id
'
------>https
host = gea.esac.esa.int:443
context = /tap-server/tap/async
Content-type = multipart/form-data; boundary====1762458134264===
303 303
[('Date', 'Thu, 06 Nov 2025 19:42:15 GMT'), ('Server', 'Apache/2.4.6 (SLES Expanded Support p
fips mod_jk/1.2.43'), ('Set-Cookie', 'JSESSIONID=7080616FCC5B47E1B29FF6FCBFFB1B25; Path=/tap-
server; Secure; HttpOnly'), ('Location', 'https://gea.esac.esa.int/tap-
server/tap/async/17624581351360'), ('Cache-Control', 'no-cache, no-store, max-
```

```
age=0, must-revalidate'), ('Pragma', 'no-cache'), ('Expires', '0'), ('X-
XSS-Protection', '1; mode=block'), ('X-Frame-Options', 'SAMEORIGIN'), ('X-
Content-Type-Options', 'nosniff'), ('Transfer-Encoding', 'chunked'), ('Content-
Type', 'text/plain;charset=ISO-8859-1')]
job 1762458135136O, at: https://gea.esac.esa.int/tap-server/tap/async/1762458135136O
Retrieving async. results...
INFO: Query finished. [astroquery.utils.tap.core]

Busca concluída!
Foram encontrados dados no DR3 para 499 das 458 estrelas.

--- 5 primeiras linhas dos membros de Taurus (dados do DR3) ---
            source_id         ra        dec  parallax       pmra      pmdec  \
0  162535413750345856  60.955653  26.181023  6.758039  20.853242 -30.277735
1  162535413754166528  60.955305  26.180727  7.803457  20.133753 -27.911397
2  162541942104406784  60.958322  26.343862  6.971044  14.435075 -19.268191
3  162535345034688768  60.961924  26.181260  7.731206  19.547795 -30.009057
4   53092775104124288  61.164068  21.971754  8.147688   3.802709 -15.247629

        ruwe
0  21.986296
1   1.159977
2   1.049943
3   2.134250
4   5.566028
```

Clustering

Carregando tabelas

```
data = pd.read_csv('taurus_membros_dr3.csv')
df = pd.DataFrame(data)
#X = df[['ra', 'dec', 'parallax', 'pmra', 'pmdec']]
#print(X['ra'])
#print(df)
# 1. Calcule as contagens de cada source_id
counts = df['source_id'].value_counts()

# 2. Use .map() para criar uma nova série do mesmo tamanho do df,
#    onde cada valor é a contagem do seu respectivo source_id.
#    Depois, compare com 2 para criar o filtro correto.
```

```
rep = df[df['source_id'].map(counts) == 2].dropna()
rep
```

	source_id	ra	dec	parallax	pmra	pmdec	ruwe
34	163179006011625088	63.717943	28.099839	7.486891	8.195294	-22.940540	1.486537
35	163179006011625216	63.718513	28.099846	7.183761	9.408134	-24.033823	0.964844
36	163179006011625088	63.717943	28.099839	7.486891	8.195294	-22.940540	1.486537
37	163179006011625216	63.718513	28.099846	7.183761	9.408134	-24.033823	0.964844
99	152416436443721728	65.289006	27.843380	7.676996	7.989072	-26.709809	1.475887
101	152416436443721728	65.289006	27.843380	7.676996	7.989072	-26.709809	1.475887
159	152180827420224128	67.177628	27.234390	7.770386	7.754825	-25.039691	1.259471
160	152180831716415488	67.177664	27.234215	7.105773	9.080319	-26.201142	1.288789
161	152180827420224128	67.177628	27.234390	7.770386	7.754825	-25.039691	1.259471
162	152180831716415488	67.177664	27.234215	7.105773	9.080319	-26.201142	1.288789
188	146361013590374144	67.623320	24.445807	6.368969	8.908057	-24.076122	2.409950
190	146361013590374144	67.623320	24.445807	6.368969	8.908057	-24.076122	2.409950
195	145921994918655488	67.709547	23.002367	8.029621	11.949303	-14.773370	3.447316
196	145921999215653632	67.709932	23.002332	6.827561	10.848324	-16.615883	1.289605
197	145921994918655488	67.709547	23.002367	8.029621	11.949303	-14.773370	3.447316
198	145921999215653632	67.709932	23.002332	6.827561	10.848324	-16.615883	1.289605
231	3314132593934981248	68.126512	17.524747	6.752014	12.825682	-19.916297	0.991326
232	3314132593936245248	68.126214	17.525034	6.873723	13.179530	-20.461011	1.109948
234	3314132593934981248	68.126512	17.524747	6.752014	12.825682	-19.916297	0.991326
235	3314132593936245248	68.126214	17.525034	6.873723	13.179530	-20.461011	1.109948
312	3313386605360382720	68.982060	17.127651	6.948271	11.938078	-19.025502	1.328659
313	3313386609655429248	68.981703	17.127255	6.674840	13.216349	-19.185871	1.189423
314	3313386605360382720	68.982060	17.127651	6.948271	11.938078	-19.025502	1.328659
315	3313386609655429248	68.981703	17.127255	6.674840	13.216349	-19.185871	1.189423
454	155744074023050368	74.714590	28.523333	6.975570	6.545382	-31.430755	0.921676
455	155744074024631296	74.714207	28.523221	6.877693	6.051624	-29.397043	1.955640
456	155744074023050368	74.714590	28.523333	6.975570	6.545382	-31.430755	0.921676
457	155744074024631296	74.714207	28.523221	6.877693	6.051624	-29.397043	1.955640
479	156430817820015232	76.956544	30.401316	5.454096	4.906034	-24.577535	16.422132
480	156430822114424576	76.956081	30.401205	6.406249	2.640643	-27.397583	1.500373
481	156430817820015232	76.956544	30.401316	5.454096	4.906034	-24.577535	16.422132
482	156430822114424576	76.956081	30.401205	6.406249	2.640643	-27.397583	1.500373
495	3415706130944329216	78.114951	22.896915	5.714827	5.981752	-18.593437	2.437345
497	3415706130944329216	78.114951	22.896915	5.714827	5.981752	-18.593437	2.437345

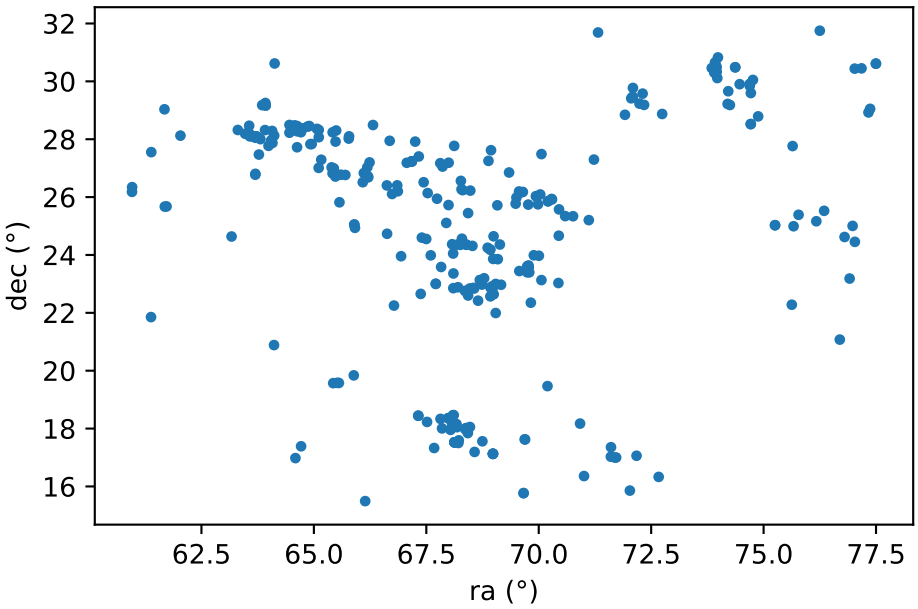
Exclui valores de RUWE <= 1.4

```
import pandas as pd

# Carregar os dados que você acabou de baixar
df_membros = pd.read_csv('taurus_membros_dr3.csv')

# Aplicar o filtro RUWE, como no artigo
df_cleaned = df_membros[df_membros['ruwe'] <= 1.4].copy()
print(f"Amostra original: {len(df_membros)} estrelas")
print(f"Amostra limpa (RUWE <= 1.4): {len(df_cleaned)} estrelas")
```

```
Amostra original: 499 estrelas
Amostra limpa (RUWE <= 1.4): 294 estrelas
```

Plota dados

```
plt.figure()
plt.plot(df_cleaned['ra'],df_cleaned['dec'], '.')
plt.xlabel('ra (°)')
plt.ylabel('dec (°)')
plt.show()
```


32
30
28
26
24
22
20
18
16
dec (°)
62.5
65.0
67.5
70.0
72.5
75.0
77.5
ra (°)