

Data Science Analysis - Assignment 5

Varenya Upadhyaya

February 18, 2023

1. The following code plots Fig. (1) containing the histograms for the densities and log of densities of asteroids from the **Penn State asteroid dataset**. We perform the Shapiro-Wilk test on both the datasets to find that the logarithm dataset gives a distribution closer to the gaussian curve than the raw data. This is apparent from the outputted p-values from the test. The best fit gaussian curves are plotted along with the histograms

```
1 import numpy as np
2 from scipy import stats
3 import matplotlib.pyplot as plt
4 import pandas as pd
5
6 df=pd.read_csv('asteroids.csv')
7 dens = df['Dens'].to_numpy()
8 logdens = np.log(dens)
9
10 #shapiro-wilk test
11 shap_dens = stats.shapiro(dens)
12 shap_logdens = stats.shapiro(logdens)
13
14 print('p-value for densities =',shap_dens.pvalue)
15 print('p-value for log of densities =',shap_logdens.pvalue)
16
17 #finding best fit gaussians
18 mu, std = stats.norm.fit(dens)
19 mu_log, std_log = stats.norm.fit(logdens)
20
21 #plotting everything
22 fig, ax = plt.subplots(1,2)
23 fig.set_figwidth(12)
24 fig.set_figheight(7)
25 fig.suptitle('Asteroid Densities')
26 x=np.linspace(-2, 6,10000)
27 ax[0].hist(dens, bins=7, density=True, label='Asteroid Densities',color='#698269')
28 ax[0].plot(x,stats.norm.pdf(x,mu,std),label= 'Best-fit gaussian',color='#B99B6B')
29 ax[0].legend()
30 ax[1].hist(logdens, bins=4, density=True, label='Log of Asteroid Densities',color
31           = '#698269')
32 ax[1].plot(x,stats.norm.pdf(x,mu_log,std_log),label= 'Best-fit gaussian',color='#B99B6B
33           ,)
34 ax[1].legend()
35 plt.savefig('1.png')
```

The output of the code :

```
p-value for densities = 0.051220282912254333
p-value for log of densities = 0.5660613775253296
```

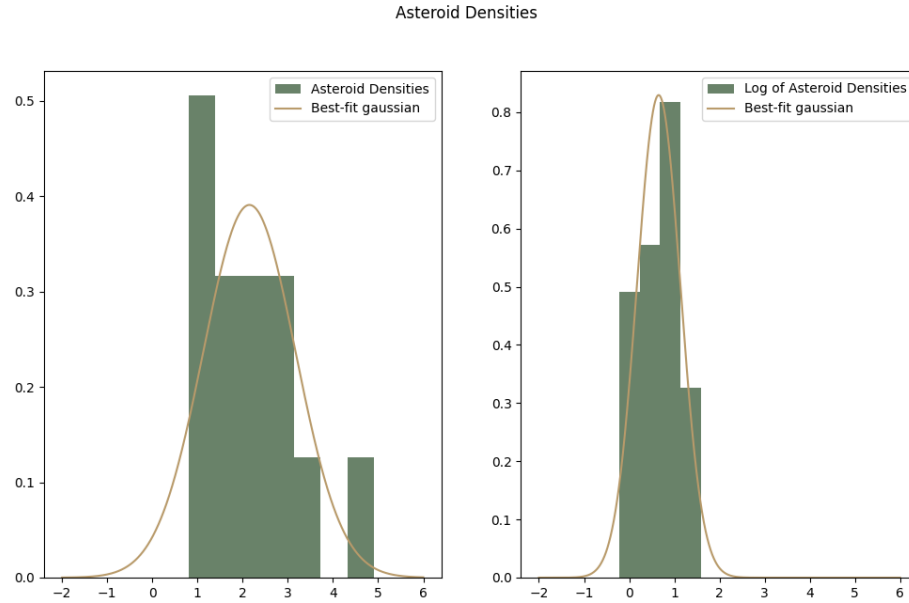


Figure 1: Histogram of the densities plotted along with the best fit gaussian curve

2. The p-value is calculated from the t-test performed on the colors of the Hyades and Non-Hyades stars in the following code. From the t-test we find a low p-value which supports the null hypothesis. The colors of the two types of stars are thus different.

```

1 import numpy as np
2 from scipy import stats
3
4 HIP, Vmag, RA, DE, Plx, pmRA, pmDE, e_Plx, BV=np.loadtxt('HIP_star.dat',skiprows=1,
5     unpack=True)
6 BV_hyades,BV_nonhyades=[],[]
7
8 for i in range(len(HIP)):
9     if (50<RA[i]<100 and 0<DE[i]<25 and 90<pmRA[i]<130 and -60<pmDE[i]<-10):
10         BV_hyades.append(BV[i])
11         continue
12     BV_nonhyades.append(BV[i])
13 t = stats.ttest_ind(BV_hyades, BV_nonhyades)
14 print('p-value =',t.pvalue)

```

Code output:

p-value = 0.00011582222192442334

3. The following code performs the GMM on the logarithm of the T90 dataset.

```

1 import numpy as np
2 from matplotlib import pyplot as plt
3 from scipy import stats
4 from sklearn.mixture import GaussianMixture
5
6 n_comp = np.arange(1,11)
7 t90 = np.loadtxt('beppoSax.txt',unpack=True)
8 data =np.log10(t90)
9 aic,bic,models=([ ] for _ in range(3))
10 for n in n_comp:
11     gmm = GaussianMixture(n_components = n)
12     X=np.expand_dims(data,1)
13     gmm = gmm.fit(X)
14     models.append(gmm)

```

```

15     aic.append(gmm.aic(X))
16     bic.append(gmm.bic(X))
17 print('Number of components =',n_comp)
18 print("AIC =", [ round(elem, 2) for elem in aic ])
19 print("BIC =", [ round(elem, 2) for elem in bic ])
20 print('opitimum n =',n_comp[np.argmin(aic)])
21
22 i_plot=[1,4,8]
23 for i in i_plot:
24     x = np.linspace(-2,4,1000)
25     y = np.exp(models[i].score_samples(x.reshape(-1,1)))
26     plt.figure(figsize=(7,10))
27     plt.plot(x, y, color="#B99B6B", lw=3, label='GMM')
28     plt.hist(data, bins=20, density=True,color='#698269')
29     plt.title('GMM for {} components'.format(n_comp[i]))
30     plt.ylabel('N')
31     plt.xlabel('log$_{10}$$(T90)$')
32     plt.legend()
33     plt.savefig('3gmm_{}'.format(str(i+1))+'.png')
34
35 #plotting the aic and bic values
36 plt.figure(figsize=(9,7))
37 plt.plot(n_comp,aic,label='AIC',ls='--',color='#B99B6B',lw=3)
38 plt.plot(n_comp,bic,label='BIC',color='#B99B6B',lw=3)
39 plt.scatter(i+1,bic[i],c='#B99B6B',s=50,)
40 plt.scatter(i+1,aic[i],c='#B99B6B',s=50)
41 plt.ylim(1840,2030)
42 plt.xlabel('number of components')
43 plt.ylabel('AIC/BIC values')
44 plt.title('AIC and BIC values for different numbers of components in the GMM')
45 plt.legend()
46 plt.savefig('3_aic.png')

```

Code output:

```

Number of components = [1  2  3  4  5  6  7  8  9 10]
AIC = [1963.18, 1851.71, 1863.22, 1867.27, 1871.14, 1868.89, 1880.13, 1877.91, 1883.78, 1890.11]
BIC = [1973.0, 1876.27, 1902.51, 1921.29, 1939.89, 1952.37, 1978.35, 1990.86, 2011.46, 2032.52]
opitimum n = 2

```

Figs. (2, 3, 4) show the best fit gaussian mixtures along with the real data. The minimum AIC and

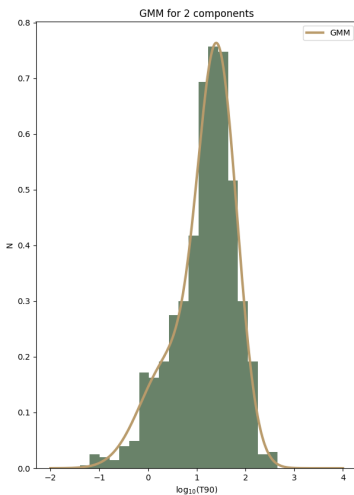


Figure 2: n=2

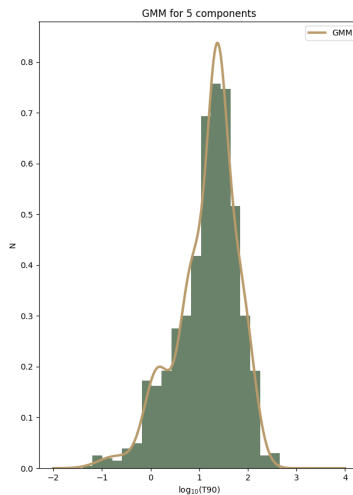


Figure 3: n=5

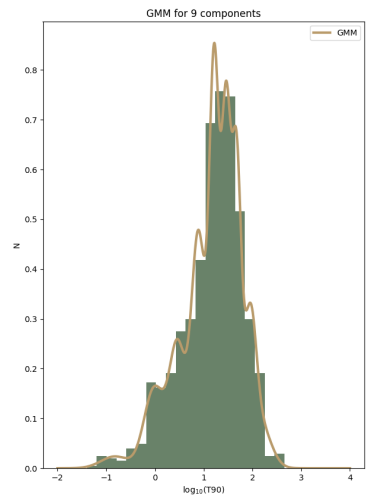


Figure 4: n=9

BIC values are attained at $n = 2$ as shown in Fig 5. Thus a mixture of 2 gaussians offers the best fit.

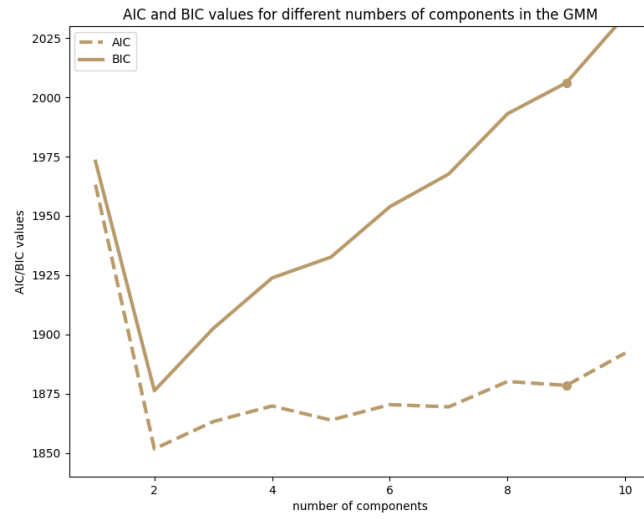


Figure 5: AIC and BIC values

All the codes and figures used in this assignment can be found [in this repository](#)