Assignment 5

Pushkal Mishra
EE20BTECH11042

Importing Libraries

[1]:
```
import numpy as np
import scipy
from scipy import stats
from scipy.stats import shapiro
from scipy.stats import norm
import matplotlib.pyplot as plt
from sklearn.mixture import GaussianMixture
```

Question 1

[2]:
```
data = np.loadtxt("q1_data.txt", delimiter = " ", dtype = str)

num = []
density = []
error = []

for line in data:
    num.append(float(line[0]))
    density.append(float(line[1]))
    error.append(float(line[2]))

num = np.array(num, dtype = "float")
density = np.array(density, dtype = "float")
log_density = np.log(density)
error = np.array(error, dtype = "float")

statistic_1, p_value_1 = shapiro(density)
statistic_2, p_value_2 = shapiro(log_density)

print(f"The results for Shapiro-Wilk test on Asteroid density-")
print(f"Statistic: {statistic_1}")
print(f"p value: {p_value_1}\n")
```

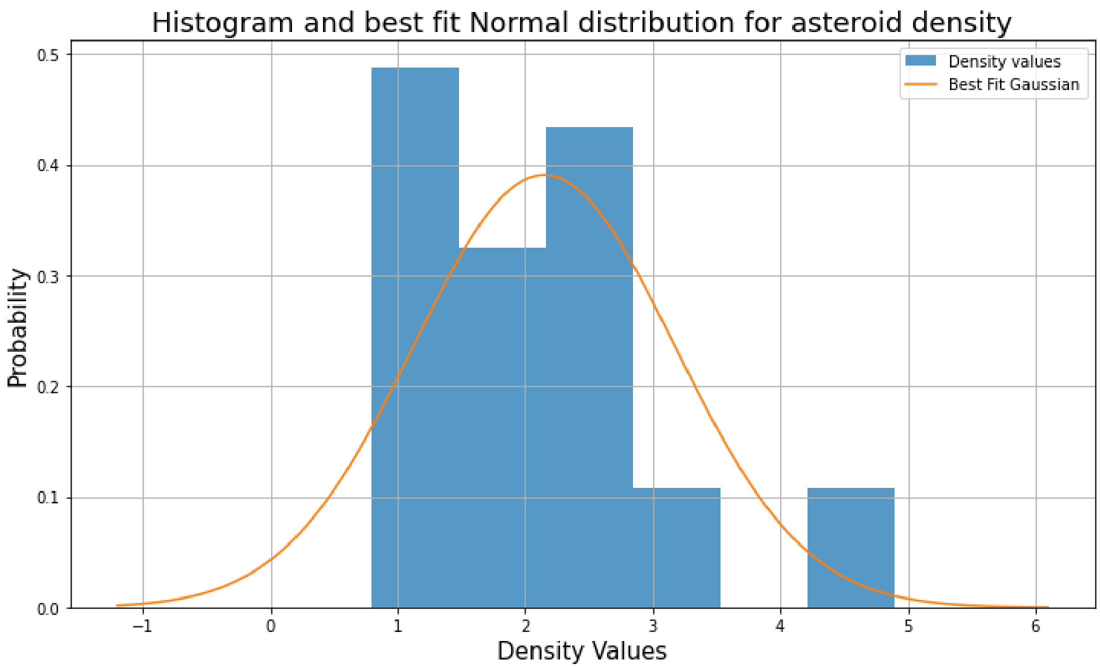
```
print(f"The results for Shapiro-Wilk test on Natural Logarithm of Asteroid␣
 ↪density-")
print(f"Statistic: {statistic_2}")
print(f"p value: {p_value_2}\n")

print(f"From the above p values, the natural log of asteroid density is closer␣
 ↪to a gaussian distribution.")
```

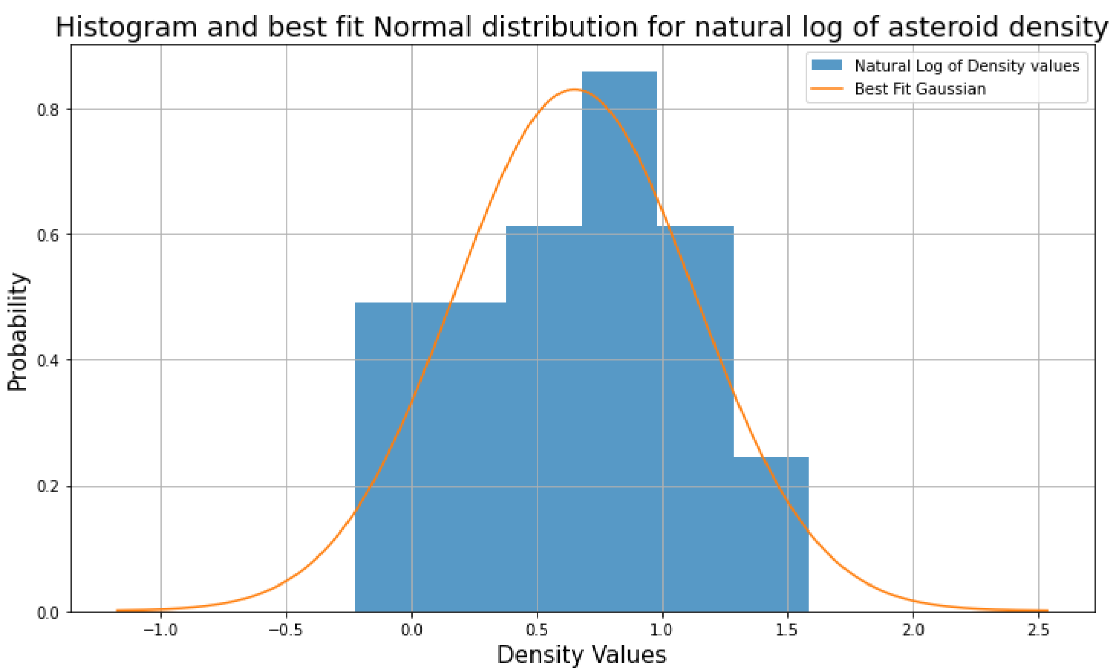
```
The results for Shapiro-Wilk test on Asteroid density-
Statistic: 0.9246721863746643
p value: 0.051220282912254333

The results for Shapiro-Wilk test on Natural Logarithm of Asteroid density-
Statistic: 0.9686306715011597
p value: 0.5660613775253296

From the above p values, the natural log of asteroid density is closer to a
gaussian distribution.
```

[3]:

```
mu_1, sigma_1 = norm.fit(density)
mu_2, sigma_2 = norm.fit(log_density)

x_plot_1 = np.linspace(np.min(density) - 2, np.max(density) + 1.2, 10000)
pdf_1 = norm(mu_1, sigma_1).pdf(x_plot_1)

x_plot_2 = np.linspace(np.min(log_density) - 0.95, np.max(log_density) + 0.95,␣
 ↪10000)
pdf_2 = norm(mu_2, sigma_2).pdf(x_plot_2)

fig = plt.figure(figsize = (12, 15))

plt.subplot(2, 1, 1)
plt.title("Histogram and best fit Normal distribution for asteroid density",␣
 ↪size = 18)
plt.hist(density, density = True, label = 'Density values', bins = 'auto', alpha␣
 ↪= 0.75)
plt.plot(x_plot_1, pdf_1, label = 'Best Fit Gaussian')
plt.xlabel("Density Values", size = 15)
plt.ylabel("Probability", size = 15)
plt.legend()
plt.grid()

plt.subplot(2, 1, 2)
plt.title("Histogram and best fit Normal distribution for natural log of␣
 ↪asteroid density", size = 18)
plt.hist(log_density, density = True, label = 'Natural Log of Density values',␣
 ↪bins = 'auto', alpha = 0.75)
```

```
plt.plot(x_plot_2, pdf_2, label = 'Best Fit Gaussian')
plt.xlabel("Density Values", size = 15)
plt.ylabel("Probability", size = 15)
plt.legend()
plt.grid()

plt.show()

print(f"Clearly normal distribution fits best for the natural logarithm of␣
↪asteroid density.")
```





Clearly normal distribution fits best for the natural logarithm of asteroid density.

Question 2

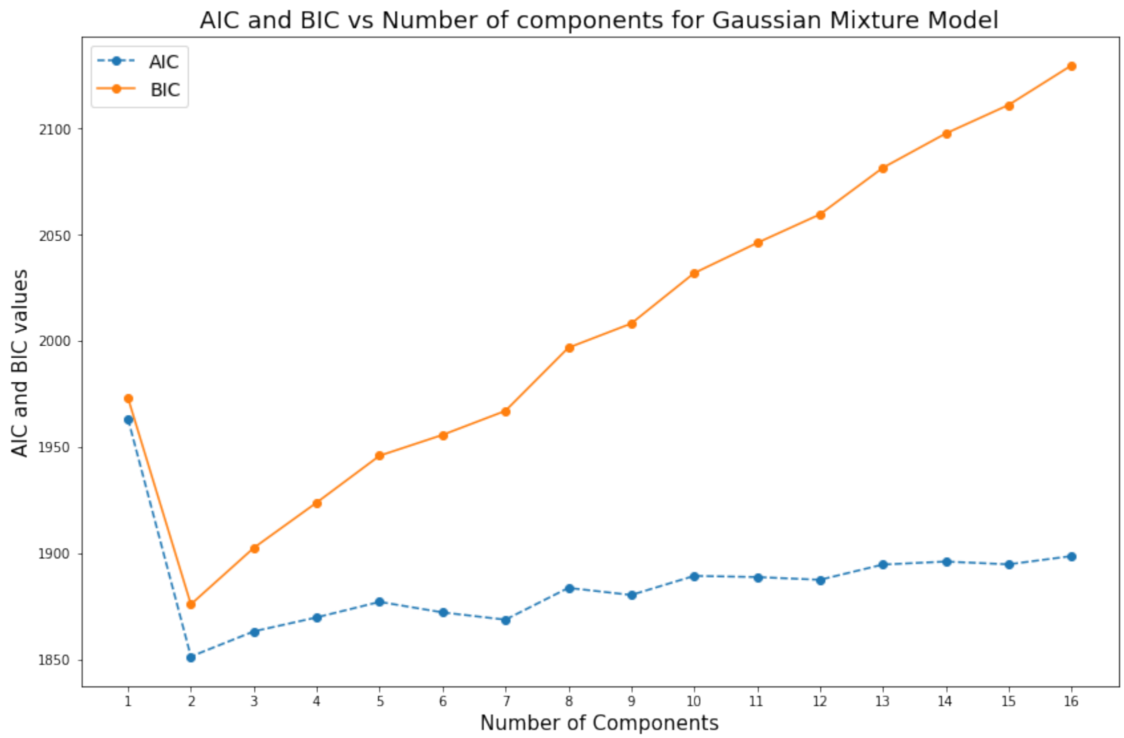
[4]:
```
data = np.loadtxt("q2_data.txt", delimiter = " ", dtype = str)

hyades_B_V = []
non_hyades_B_V = []

for i in range(1, len(data)):
    RA = float(data[i][2])
    DE = float(data[i][3])
    pmRA = float(data[i][5])
    pmDE = float(data[i][6])
    B_V = float(data[i][8])

    if RA>=50 and RA<=100 and DE>=0 and DE<=25 and pmRA>=90 and pmRA<=130 and␣
 ↪pmDE>=-60 and pmDE<=-10:
        hyades_B_V.append(B_V)
    else:
        non_hyades_B_V.append(B_V)

hyades_B_V = np.array(hyades_B_V, dtype = "float")
non_hyades_B_V = np.array(non_hyades_B_V, dtype = "float")

print(f"Number of Hyades stars: {len(hyades_B_V)}")
print(f"Number of Non-Hyades stars: {len(non_hyades_B_V)}\n")

print(f"Number of Non-Hyades stars is clearly much more than number of Hyades␣
 ↪stars.\n")

var_hyades = np.var(hyades_B_V)
var_non_hyades = np.var(non_hyades_B_V)

print(f"Ratio of variances of Non-Hyades to Hyades stars: {var_non_hyades /␣
 ↪var_hyades}")
print("Since the above ratio is less than 4:1, we can consider the data groups␣
 ↪to have equal variances.\n")

statstic, p_value = stats.ttest_ind(a = hyades_B_V, b = non_hyades_B_V,␣
 ↪equal_var = True)

print("The results of t-test are as follows-")
print(f"Statistic: {statstic}")
print(f"p value: {p_value}\n")

print("This small p value indicates that the color of Hyades stars differs from␣
 ↪the color of Non-Hyades stars.")
```

```
Number of Hyades stars: 93
Number of Non-Hyades stars: 2626

Number of Non-Hyades stars is clearly much more than number of Hyades stars.

Ratio of variances of Non-Hyades to Hyades stars: 1.018601840454133
Since the above ratio is less than 4:1, we can consider the data groups to have
equal variances.

The results of t-test are as follows-
Statistic: -3.860436921860911
p value: 0.00011582222192442334

This small p value indicates that our assumption that both color of Hyades and
Non-Haydes stars are the same is False.
So the color Hyades and Non-Hyades starrs are different.
```

Question 3

[5]:

```
data = np.loadtxt("q3_data.txt", dtype = float)
log_data = np.log10(data).reshape([len(data), 1])

num_components = np.arange(1, 17, dtype = int)
AIC = []
BIC = []

for num in num_components:
    model = GaussianMixture(n_components = num, covariance_type = 'full',␣
 ↪max_iter = 1000)
    fit_model = model.fit(log_data)

    AIC.append(fit_model.aic(log_data))
    BIC.append(fit_model.bic(log_data))

fig = plt.figure(figsize = (14, 9))

plt.title("AIC and BIC vs Number of components for Gaussian Mixture Model", size␣
 ↪= 18)
plt.plot(num_components, AIC, label = 'AIC', ls = '--', marker = 'o')
plt.plot(num_components, BIC, label = 'BIC', marker = 'o')
plt.xlabel("Number of Components", size = 15)
plt.ylabel("AIC and BIC values", size = 15)
plt.xticks(num_components)
plt.legend(fontsize = 14)

plt.show()
```



[6]:
```
print("Clearly from the above plot, lowest AIC and BIC values are obtained at 2␣
↪components.")
print("Therefore the optimal number of Gaussian Components for the given data is␣
↪2.")
```

```
Clearly from the above plot, lowest AIC and BIC values are obtained at 2
components.
Therefore the optimal number of Gaussian Components for the given data is 2.
```