Elif KARTAL

Bu ödev Jupyter Notbook aracılığı ile python kodları ile yazılmıştır. Kullanılan veriler algoritmanın doğruluğunu teyit etmek amacı ile derste örneği çözülen veri seti kullanılmıştır. Veri 3 grup 2 değişkenden oluşmaktadır. Veri Kronik hepatit, siroz ve Malignite hastalarının kanlarındaki Beta ve Albumin değerlerinin gözlenmesi ile oluşturulmuştur.

Veri setinin tanımlanması:

In [84]:
```
import numpy as np
import pandas as pd
veriler = {
    "Grup": ["Kronik Hepatit"] * 15 + ["Siroz"] * 15 + ["Malignite"] * 15,
    "Beta": [21,16,15,19,18,13,16,14,17,16,19,15,17,17,16] +
             [11,13,14,10,9,12,10,8,12,11,14,9,13,11,10] +
             [17,15,17,18,17,16,14,16,17,15,19,19,14,16,13],
    "Albumin": [5.0, 5.1, 4.5, 4.7, 2.8, 5.3, 4.7, 4.5, 3.6, 3.8, 2.7, 3.8, 3.5, 5.
                [3.0,4.3,3.4,1.8,2.2,2.7,2.5,3.1,2.8,2.2,1.5,1.6,1.7,1.4,1.9] +
                [0.8,1.3,2.2,2.7,1.9,1.4,2.6,1.0,1.5,0.7,0.4,1.4,1.5,1.7,1.0]
}
df = pd.DataFrame(veriler)
```

Çalışmanın devamında kullanacağımız parametrelerin hesaplanması:

In [94]:
```
# Değerlerin hesaplanması
N = df.shape[0]  # Toplam gözlem sayısı (satır sayısı)
p = len([col for col in df.columns if col != "Grup"])  # Bağımlı değişken sayısı
k = df["Grup"].nunique()  # Farklı grup sayısı
alpha = 0.05 #Anlamlılık seviyesi
df1 = 2 * (k - 1) # serbestlik derecesi
df2 = 2 * (N - p-1)  # serbestlik derecesi


# Sonuçların yazdırılması
print(f"N (Gözlem Sayısı): {N}")
print(f"P (Bağımlı Değişken Sayısı): {P}")
print(f"K (Grup Sayısı): {K}")
print(f"alpha (Anlamlılık Seviyesi): {alpha}")
print(f"df1 (Serbestlik Derecesi): {df1}")
print(f"df2 (Serbestlik Derecesi): {df2}")
```

```
N (Gözlem Sayısı): 45
P (Bağımlı Değişken Sayısı): 2
K (Grup Sayısı): 3
alpha (Anlamlılık Seviyesi): 0.05
df1 (Serbestlik Derecesi): 4
df2 (Serbestlik Derecesi): 84
```

Veri setinin görünümü:

In [86]:
```
df
```

Out[86]:

	Grup	Beta	Albumin
0	Kronik Hepatit	21	5.0
1	Kronik Hepatit	16	5.1
2	Kronik Hepatit	15	4.5
3	Kronik Hepatit	19	4.7
4	Kronik Hepatit	18	2.8
5	Kronik Hepatit	13	5.3
6	Kronik Hepatit	16	4.7
7	Kronik Hepatit	14	4.5
8	Kronik Hepatit	17	3.6
9	Kronik Hepatit	16	3.8
10	Kronik Hepatit	19	2.7
11	Kronik Hepatit	15	3.8
12	Kronik Hepatit	17	3.5
13	Kronik Hepatit	17	5.0
14	Kronik Hepatit	16	3.1
15	Siroz	11	3.0
16	Siroz	13	4.3
17	Siroz	14	3.4
18	Siroz	10	1.8
19	Siroz	9	2.2
20	Siroz	12	2.7
21	Siroz	10	2.5
22	Siroz	8	3.1
23	Siroz	12	2.8
24	Siroz	11	2.2
25	Siroz	14	1.5
26	Siroz	9	1.6
27	Siroz	13	1.7
28	Siroz	11	1.4
29	Siroz	10	1.9

	Grup	Beta	Albumin
30	Malignite	17	0.8
31	Malignite	15	1.3
32	Malignite	17	2.2
33	Malignite	18	2.7
34	Malignite	17	1.9
35	Malignite	16	1.4
36	Malignite	14	2.6
37	Malignite	16	1.0
38	Malignite	17	1.5
39	Malignite	15	0.7
40	Malignite	19	0.4
41	Malignite	19	1.4
42	Malignite	14	1.5
43	Malignite	16	1.7
44	Malignite	13	1.0

Bağımlı değişkenlerin atamaları ve ortalamalarının hesaplanması:

In [91]:
```
# Grup etiketleri ve bağımlı değişkenlerin belirlenmesi
group_labels = df["Grup"].unique()
dependent_vars = ["Beta", "Albumin"]

# Her grup için ortalamalar
means = df.groupby("Grup")[dependent_vars].mean()
overall_mean = df[dependent_vars].mean()

# İçe Grubu Toplam (Within Group) ve Grup Dışı Toplam (Between Group) Matrisleri
within_group = np.zeros((len(dependent_vars), len(dependent_vars)))
between_group = np.zeros((len(dependent_vars), len(dependent_vars)))
```

In [89]:
```
means #grupların ortalamaları
```

Out[89]:

Grup	Beta	Albumin
Kronik Hepatit	16.600000	4.140000
Malignite	16.200000	1.473333
Siroz	11.133333	2.406667

In [7]:
```
overall_mean #genel ortalama
```

Out[7]:

	0
Beta	14.644444
Albumin	2.673333

dtype: float64

In [8]:
```
within_group #grup içi kareler toplamı matrisi
```

Out[8]:
```
array([[151.73333333,  -1.69333333],
       [ -1.69333333,  26.09466667]])
```

In [9]:
```
between_group #gruplar arası kareler toplamı matrisi
```

Out[9]:
```
array([[278.57777778,  29.06666667],
       [ 29.06666667,  54.93333333]])
```

In [10]:
```
within_group + between_group #grup içi ve gruplar arası kareler toplamı matrislerin
```

Out[10]:
```
array([[430.31111111,  27.37333333],
       [ 27.37333333,  81.028     ]])
```

Grupların normal dağılımı varsayımının shapiro wilk testi ile kontrol edilmesi:

In [92]:
```
from scipy.stats import shapiro

grouped = df.groupby("Grup")

print("Gruplar için normallik testi sonuçları:")
print("-" * 50)

for group_name, group_data in grouped:
    # Beta değişkeni için normallik testi
    stat_beta, p_beta = shapiro(group_data["Beta"])
    # Albumin değişkeni için normallik testi
    stat_albumin, p_albumin = shapiro(group_data["Albumin"])

    # Sonuçları yazdır
    print(f"Grup: {group_name}")
    print(f"Beta - Test İstatistiği: {stat_beta:.4f}, P-Değeri: {p_beta:.4f}")
```

```
    if p_beta < 0.05:
        print("Beta değişkeni normal dağılıma uymuyor.")
    else:
        print("Beta değişkeni normal dağılıma uyuyor.")

    print(f"Albumin - Test İstatistiği: {stat_albumin:.4f}, P-Değeri: {p_albumin:.4
    if p_albumin < 0.05:
        print("Albumin değişkeni normal dağılıma uymuyor.")
    else:
        print("Albumin değişkeni normal dağılıma uyuyor.")

    print("-" * 50)
```

```
Gruplar için normallik testi sonuçları:
--------------------------------------------------
Grup: Kronik Hepatit
Beta - Test İstatistiği: 0.9694, P-Değeri: 0.8483
Beta değişkeni normal dağılıma uyuyor.
Albumin - Test İstatistiği: 0.9219, P-Değeri: 0.2062
Albumin değişkeni normal dağılıma uyuyor.
--------------------------------------------------
Grup: Malignite
Beta - Test İstatistiği: 0.9570, P-Değeri: 0.6411
Beta değişkeni normal dağılıma uyuyor.
Albumin - Test İstatistiği: 0.9617, P-Değeri: 0.7224
Albumin değişkeni normal dağılıma uyuyor.
--------------------------------------------------
Grup: Siroz
Beta - Test İstatistiği: 0.9549, P-Değeri: 0.6039
Beta değişkeni normal dağılıma uyuyor.
Albumin - Test İstatistiği: 0.9364, P-Değeri: 0.3397
Albumin değişkeni normal dağılıma uyuyor.
--------------------------------------------------
```

Varyans-Kovaryans Matrisleri

Gruplar için varyans kovaryans matrislerinin hesaplanması:

In [68]:
```
# Her grup için varyans-kovaryans matrisini hesapla
grouped = df.groupby("Grup")  # Gruplara ayırmak için

print("Her grup için varyans-kovaryans matrisleri:\n")

for group_name, group_data in grouped:
    cov_matrix = group_data[["Beta", "Albumin"]].cov()
    print(f"Grup: {group_name}")
    print(cov_matrix)
    print("-" * 30)
```

```
Her grup için varyans-kovaryans matrisleri:

Grup: Kronik Hepatit
              Beta   Albumin
Beta     4.257143 -0.397143
Albumin -0.397143  0.751143
------------------------------
Grup: Malignite
              Beta   Albumin
Beta     3.171429 -0.030000
Albumin -0.030000  0.444952
------------------------------
Grup: Siroz
              Beta  Albumin
Beta     3.409524  0.30619
Albumin  0.306190  0.66781
------------------------------
```

Manova varsayımlarından varyans homojenliğinin Box's M testi ile kontrol edilmesi:

In [78]:

```
from scipy.stats import chi2
import numpy as np

# Grupların varyans-kovaryans matrislerini hesaplayalım
grouped = df.groupby("Grup")
cov_matrices = [group_data[["Beta", "Albumin"]].cov().values for _, group_data in g

# Grupların gözlem sayılarını al
n = [len(group_data) for _, group_data in grouped]
N = sum(n)  # Toplam gözlem sayısı
g = len(cov_matrices)  # Grup sayısı
p = cov_matrices[0].shape[0]  # Değişken sayısı

# Pooled kovaryans matrisini hesapla
pooled_cov = sum([(n_i - 1) * cov for n_i, cov in zip(n, cov_matrices)]) / (N - g)

# Determinantları hesapla
ln_determinants = [(n_i - 1) * np.log(np.linalg.det(cov)) for n_i, cov in zip(n, co
ln_pooled = (N - g) * np.log(np.linalg.det(pooled_cov))

# Box's M istatistiğini hesapla
M = sum(ln_determinants) - ln_pooled

# Düzeltme faktörü
C = (2 * p**2 + 3 * p - 1) * (g - 1) / (6 * (N - g))

# Düzeltilmiş Box's M testi istatistiği
M_prime = M * (1 - C)

# Serbestlik derecesi
sd = (g - 1) * p * (p + 1) / 2

# P-değerini hesapla
p_value = 1 - chi2.cdf(M_prime, sd)

print("Box's M Testi:")
```

```
print(f"M' İstatistiği: {M_prime:.4f}")
print(f"Serbestlik Derecesi: {sd:.0f}")
print(f"P-değeri: {p_value:.4f}")

# Yorum
if p_value > 0.05:
    print("Sonuç: Varyans-kovaryans matrisleri homojendir.MANOVA testi güvenle uygu
else:
    print("Sonuç: Grupların varyans-kovaryans matrisleri homojen değildir. MANOVA U
```

```
Box's M Testi:
M' İstatistiği: -2.3477
Serbestlik Derecesi: 6
P-değeri: 1.0000
Sonuç: Varyans-kovaryans matrisleri homojendir.MANOVA testi güvenle uygulanabilir.
```

MANOVA

H0:μ 1=μ 2=μ 3=...=μ k

H1:En az bir μ i, diğerlerinden farklıdır

1.Wilks' Lambda İstatistiği

In [11]:
```
# Wilks' Lambda Hesaplama
wilks_lambda = np.linalg.det(within_group) / np.linalg.det(within_group + between_g
print(f"Wilks' Lambda: {wilks_lambda:.4f}")
```

```
Wilks' Lambda: 0.1160
```

In [12]:
```
import scipy.stats as stats
import math
wilks_lambda = wilks_lambda  #  Wilks' Lambda değeri

# F istatistiğini hesapla
f_statistic = ((n- p - 2) / (p)) * (((1 - math.sqrt(wilks_lambda))) / math.sqrt(wil

# Kritik değeri hesapla
critical_value = stats.f.ppf(1-alpha, 2*(k-1), 2*(n-k-1))

# Sonuçları yazdır
print(f"F İstatistiği: {f_statistic:.2f}")
print(f"Kritik Değer: {critical_value:.2f}")

# Yorum
if f_statistic > critical_value:
    print("Sonuç: Anlamlı, gruplar arasında farklılık vardır.")
else:
    print("Sonuç: Anlamlı değil, gruplar arasında farklılık yoktur.")
```

```
F İstatistiği: 39.70
Kritik Değer: 2.48
Sonuç: Anlamlı, gruplar arasında farklılık vardır.
```

2. Hotelling İz İstatistiği

In [13]:
```
import numpy as np
#özdeğerlerin hesaplanması
eigenvalues = np.linalg.eigvals(np.linalg.inv(within_group).dot(between_group))

hotellings_trace = sum(eigenvalues)  # Özdeğerlerin toplamı
print(f"Hotelling’s Trace: {hotellings_trace:.4f}")
```

```
Hotelling’s Trace: 3.9689
```

In [14]:
```
eigenvalues #özdeğerlerin yazdırılması
```

Out[14]:
```
array([1.45179662, 2.51706465])
```

In [15]:
```
import scipy.stats as stats

df2 = 4
hotellings_trace = hotellings_trace  # Hotelling's Trace değeri

# Ki-kare istatistiğini hesapla
chi_square_statistic = n * hotellings_trace

# Kritik değeri hesapla
critical_value = stats.chi2.ppf( 1-alpha, df2)

# Sonuçları yazdır
print(f"Ki-Kare İstatistiği: {chi_square_statistic:.4f}")
print(f"Kritik Değer: {critical_value:.4f}")

# Yorum
if chi_square_statistic > critical_value:
    print("Sonuç: Anlamlı, gruplar arasında farklılık vardır.")
else:
    print("Sonuç: Anlamlı değil, gruplar arasında farklılık yoktur.")
```

```
Ki-Kare İstatistiği: 178.5988
Kritik Değer: 9.4877
Sonuç: Anlamlı, gruplar arasında farklılık vardır.
```

3. Pillai'nin İz İstatistiği

In [16]:
```
# Pillai's Trace Hesaplama
pillais_trace = sum(eigenvalues / (1 + eigenvalues))
print(f"Pillai's Trace: {pillais_trace:.4f}")
```

```
Pillai's Trace: 1.3078
```

In [50]:
```
import scipy.stats as stats

pillais_trace = pillais_trace  # Hesaplanan Pillai's Trace
```

```
# Hesaplamalar
s = min(p, k - 1)
m = (abs(s - (k - 1)) - 1) / 2
a = (N - k - p - 1) / 2

# F-istatistiğini hesaplama
f_stat = ((2 * a + s + 1) / (2 * m + s + 1)) * (pillais_trace / (s - pillais_trace)

# Kritik değer ve p-değeri
critical_value = stats.f.ppf(1 - alpha, df1, df2)
p_value = 1 - stats.f.cdf(f_stat, df1, df2)

# Sonuçların yazdırılması
print(f"F-Statistiği: {f_stat:.4f}")
print(f"Kritik Değer: {critical_value:.4f}")
print(f"p-Değeri: {p_value:.4f}")

# Sonucun yorumu
if f_stat > critical_value:
    print("Sonuç: Grup ortalamaları arasında anlamlı bir fark vardır (H0 reddedilir
else:
    print("Sonuç: Grup ortalamaları arasında anlamlı bir fark yoktur (H0 reddedilem
```

```
F-Statistiği: 39.6761
Kritik Değer: 2.4803
p-Değeri: 0.0000
Sonuç: Grup ortalamaları arasında anlamlı bir fark vardır (H0 reddedilir).
```

4. Roy'un En Büyük Özdeğere Dayalı Test İstatistiği

In [58]:
```
# Roy’s Largest Root Hesaplama
roys_largest_root = max(eigenvalues) / (1+(max(eigenvalues)))
print(f"Roy’s Largest Root: {roys_largest_root:.4f}")
```

```
Roy’s Largest Root: 0.7157
```

In [60]:
```
import scipy.stats as stats

roys_largest_root = 0.7157  # Roy's Largest Root değeri

f_statistic = ((N - k) * roys_largest_root) / (k - 1)

# Kritik değeri hesapla
critical_value = stats.f.ppf(1-alpha, df1, df2)

# Sonuçları yazdır
print(f"F İstatistiği: {f_statistic:.4f}")
print(f"Kritik Değer: {critical_value:.4f}")

# Yorum
if f_statistic > critical_value:
    print("Sonuç: Anlamlı, gruplar arasında farklılık vardır.")
```

```
else:
    print("Sonuç: Anlamlı değil, gruplar arasında farklılık yoktur.")
```

```
F İstatistiği: 15.0291
Kritik Değer: 3.9546
Sonuç: Anlamlı, gruplar arasında farklılık vardır.
```

Bonferroni Eş Zamanlı Güven Aralıklarını Hesaplama

Gruplar arası farkın tespiti için Benferroni eş zamanlı güven aralıklarını hesaplama:

In [133…

```
# Bonferroni eş zamanlı güven aralıklarını hesaplama
alpha = 0.05  # Anlamlılık seviyesi
n_groups = len(group_labels)  # Grup sayısı
n_total = len(df)  # Toplam veri sayısı

results = []  # Sonuçlar için liste

# Tüm bağımlı değişkenler için hesaplama
for var in dependent_vars:
    group_means = df.groupby('Grup')[var].mean()  # Grup ortalamaları
    group_counts = df.groupby('Grup').size()  # Her gruptaki örnek sayısı
    pooled_var = df.groupby('Grup')[var].var().mean()  # Ortak varyans

    # Bonferroni düzeltmesi için t-kritik değeri
    t_critical = t.ppf(1 - alpha / (n_groups * (n_groups - 1) * 2), df=n_total - n_

    # Gruplar arasındaki farklar ve güven aralıklarını hesapla
    for (group1, group2) in combinations(group_labels, 2):
        mean_diff = group_means[group1] - group_means[group2]
        se_diff = np.sqrt(pooled_var * (1 / group_counts[group1] + 1 / group_counts
        margin_of_error = t_critical * se_diff
        lower_bound = mean_diff - margin_of_error
        upper_bound = mean_diff + margin_of_error
        results.append({
            'Variable': var,
            'Group1': group1,
            'Group2': group2,
            'Lower Bound': lower_bound,
            'Upper Bound': upper_bound,
            'Significant': (lower_bound > 0) or (upper_bound < 0)  # Güven aralığı
        })

# Sonuçları yazdır
results_df = pd.DataFrame(results)
print("\nBonferroni Eş Zamanlı Güven Aralıkları:")
print(results_df)
```

```
Bonferroni Eş Zamanlı Güven Aralıkları:
  Variable          Group1     Group2  Lower Bound  Upper Bound  Significant
0     Beta  Kronik Hepatit      Siroz     3.544856     7.388478         True
1     Beta  Kronik Hepatit  Malignite    -1.521811     2.321811        False
2     Beta           Siroz  Malignite    -6.988478    -3.144856         True
3  Albumin  Kronik Hepatit      Siroz     0.936356     2.530311         True
4  Albumin  Kronik Hepatit  Malignite     1.869689     3.463644         True
5  Albumin           Siroz  Malignite     0.136356     1.730311         True
```

Sonuçların İncelenmesi:

Beta Değişkeni:

- Kronik Hepatit vs. Siroz:

Güven aralığı: [3.544856 , 7.388478] [3.544856,7.388478] Güven aralığı 0'ı içermiyor → Fark anlamlı. Yorum: Beta değerlerinde Kronik Hepatit ve Siroz grupları arasında istatistiksel olarak anlamlı bir fark var.

- Kronik Hepatit vs. Malignite:

Güven aralığı: [− 1.521811 , 2.321811] [−1.521811,2.321811] Güven aralığı 0'ı içeriyor → Fark anlamlı değil. Yorum: Beta değerlerinde Kronik Hepatit ve Malignite grupları arasında anlamlı bir fark yok.

- Siroz vs. Malignite:

Güven aralığı: [− 6.988478 , − 3.144856] [−6.988478,−3.144856] Güven aralığı 0'ı içermiyor → Fark anlamlı. Yorum: Beta değerlerinde Siroz ve Malignite grupları arasında istatistiksel olarak anlamlı bir fark var.

Albumin Değişkeni:

- Kronik Hepatit vs. Siroz:

Güven aralığı: [0.936356 , 2.530311] [0.936356,2.530311] Güven aralığı 0'ı içermiyor → Fark anlamlı. Yorum: Albumin değerlerinde Kronik Hepatit ve Siroz grupları arasında istatistiksel olarak anlamlı bir fark var.

- Kronik Hepatit vs. Malignite:

Güven aralığı: [1.869689 , 3.463644] [1.869689,3.463644] Güven aralığı 0'ı içermiyor → Fark anlamlı. Yorum: Albumin değerlerinde Kronik Hepatit ve Malignite grupları arasında istatistiksel olarak anlamlı bir fark var.

- Siroz vs. Malignite:

Güven aralığı: [0.136356 , 1.730311] [0.136356,1.730311] Güven aralığı 0'ı içermiyor → Fark anlamlı. Yorum: Albumin değerlerinde Siroz ve Malignite grupları arasında istatistiksel olarak anlamlı bir fark var.