2021101113-pcafa-class-activity

March 21, 2024

0.1 [1] Wine Dataset

[62]:
```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.decomposition import PCA
from scipy.stats import chisquare
from sklearn.preprocessing import StandardScaler , MinMaxScaler
from scipy.stats import chi2_contingency
```

[63]:
```
df = pd.read_csv('wine.csv')
original_data = df
X = df.drop(columns=['Type'])
y = df['Type']
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
df_scaled = pd.DataFrame(X_scaled, columns=X.columns)
df = df_scaled
print(df.columns)
print(df.describe().to_string())
print(y.value_counts())
```

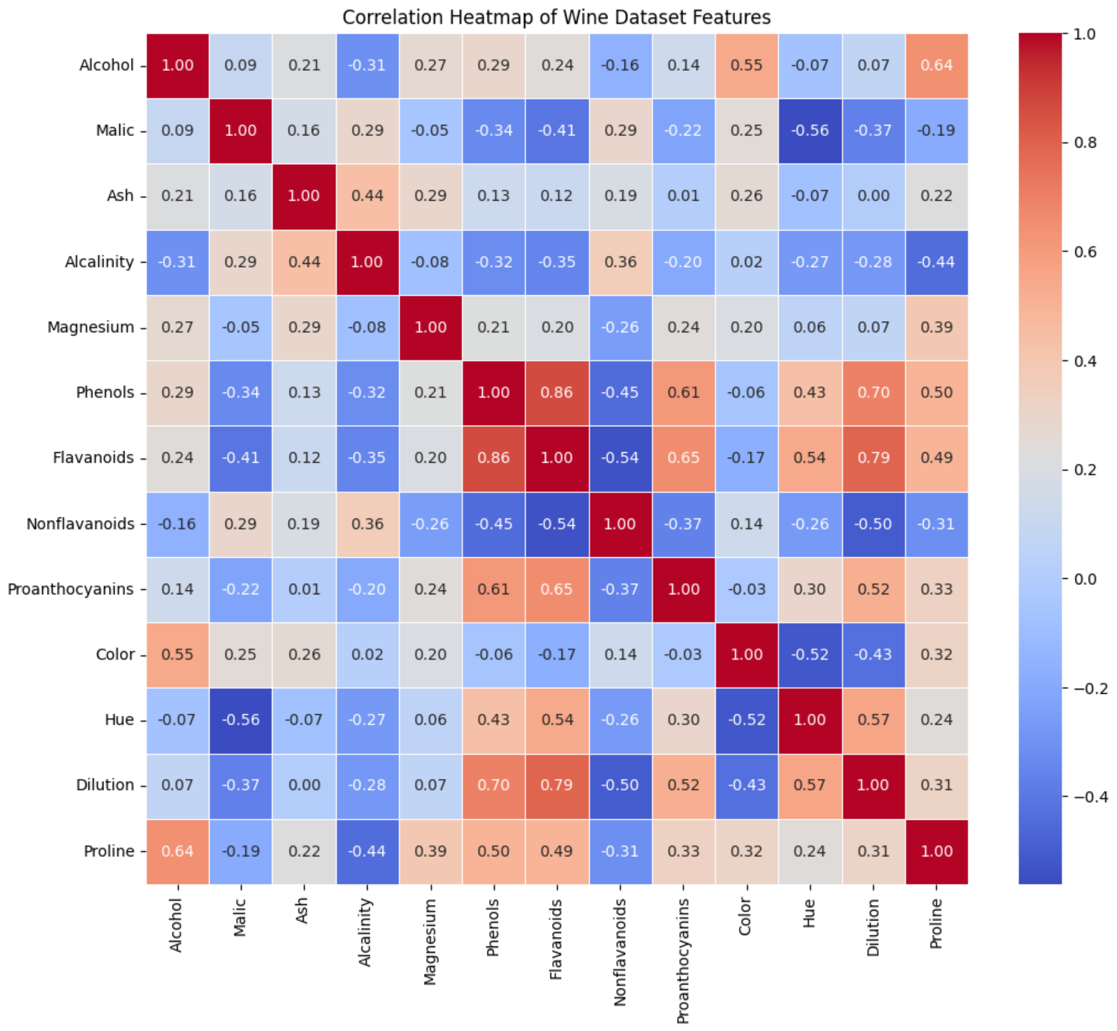
```
Index(['Alcohol', 'Malic', 'Ash', 'Alcalinity', 'Magnesium', 'Phenols',
       'Flavanoids', 'Nonflavanoids', 'Proanthocyanins', 'Color', 'Hue',
       'Dilution', 'Proline'],
      dtype='object')
            Alcohol          Malic            Ash     Alcalinity      Magnesium
Phenols     Flavanoids  Nonflavanoids  Proanthocyanins          Color
Hue       Dilution        Proline
count   1.780000e+02   1.780000e+02   1.780000e+02   1.780000e+02   1.780000e+02
178.000000   1.780000e+02    1.780000e+02      1.780000e+02   1.780000e+02
1.780000e+02   1.780000e+02   1.780000e+02
mean   -8.382808e-16 -1.197544e-16 -8.370333e-16 -3.991813e-17 -3.991813e-17
0.000000 -3.991813e-16    3.592632e-16    -1.197544e-16   2.494883e-17
1.995907e-16   3.193450e-16 -1.596725e-16
std     1.002821e+00   1.002821e+00   1.002821e+00   1.002821e+00   1.002821e+00
1.002821   1.002821e+00    1.002821e+00      1.002821e+00   1.002821e+00
```

```
1.002821e+00  1.002821e+00  1.002821e+00
min   -2.434235e+00 -1.432983e+00 -3.679162e+00 -2.671018e+00 -2.088255e+00
-2.107246 -1.695971e+00  -1.868234e+00    -2.069034e+00 -1.634288e+00
-2.094732e+00 -1.895054e+00 -1.493188e+00
25%   -7.882448e-01 -6.587486e-01 -5.721225e-01 -6.891372e-01 -8.244151e-01
-0.885468 -8.275393e-01  -7.401412e-01    -5.972835e-01 -7.951025e-01
-7.675624e-01 -9.522483e-01 -7.846378e-01
50%    6.099988e-02 -4.231120e-01 -2.382132e-02  1.518295e-03 -1.222817e-01
0.095960  1.061497e-01  -1.760948e-01    -6.289785e-02 -1.592246e-01
3.312687e-02  2.377348e-01 -2.337204e-01
75%    8.361286e-01  6.697929e-01  6.981085e-01  6.020883e-01  5.096384e-01
0.808997  8.490851e-01   6.095413e-01     6.291754e-01  4.939560e-01
7.131644e-01  7.885875e-01  7.582494e-01
max    2.259772e+00  3.109192e+00  3.156325e+00  3.154511e+00  4.371372e+00
2.539515  3.062832e+00   2.402403e+00     3.485073e+00  3.435432e+00
3.301694e+00  1.960915e+00  2.971473e+00
Type
2    71
1    59
3    48
Name: count, dtype: int64
```

0.2 [2] Principal Component Analysis (PCA)

0.2.1 [2.1] Correlation heatmap of features

[64]:
```
correlation_matrix = df.corr()
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f",␣
 ↪linewidths=0.5)
plt.title('Correlation Heatmap of Wine Dataset Features')
plt.show()
```



	Alcohol	Malic	Ash	Alcalinity	Magnesium	Phenols	Flavanoids	Nonflavanoids	Proanthocyanins	Color	Hue	Dilution	Proline
Alcohol	1.00	0.09	0.21	-0.31	0.27	0.29	0.24	-0.16	0.14	0.55	-0.07	0.07	0.64
Malic	0.09	1.00	0.16	0.29	-0.05	-0.34	-0.41	0.29	-0.22	0.25	-0.56	-0.37	-0.19
Ash	0.21	0.16	1.00	0.44	0.29	0.13	0.12	0.19	0.01	0.26	-0.07	0.00	0.22
Alcalinity	-0.31	0.29	0.44	1.00	-0.08	-0.32	-0.35	0.36	-0.20	0.02	-0.27	-0.28	-0.44
Magnesium	0.27	-0.05	0.29	-0.08	1.00	0.21	0.20	-0.26	0.24	0.20	0.06	0.07	0.39
Phenols	0.29	-0.34	0.13	-0.32	0.21	1.00	0.86	-0.45	0.61	-0.06	0.43	0.70	0.50
Flavanoids	0.24	-0.41	0.12	-0.35	0.20	0.86	1.00	-0.54	0.65	-0.17	0.54	0.79	0.49
Nonflavanoids	-0.16	0.29	0.19	0.36	-0.26	-0.45	-0.54	1.00	-0.37	0.14	-0.26	-0.50	-0.31
Proanthocyanins	0.14	-0.22	0.01	-0.20	0.24	0.61	0.65	-0.37	1.00	-0.03	0.30	0.52	0.33
Color	0.55	0.25	0.26	0.02	0.20	-0.06	-0.17	0.14	-0.03	1.00	-0.52	-0.43	0.32
Hue	-0.07	-0.56	-0.07	-0.27	0.06	0.43	0.54	-0.26	0.30	-0.52	1.00	0.57	0.24
Dilution	0.07	-0.37	0.00	-0.28	0.07	0.70	0.79	-0.50	0.52	-0.43	0.57	1.00	0.31
Proline	0.64	-0.19	0.22	-0.44	0.39	0.50	0.49	-0.31	0.33	0.32	0.24	0.31	1.00

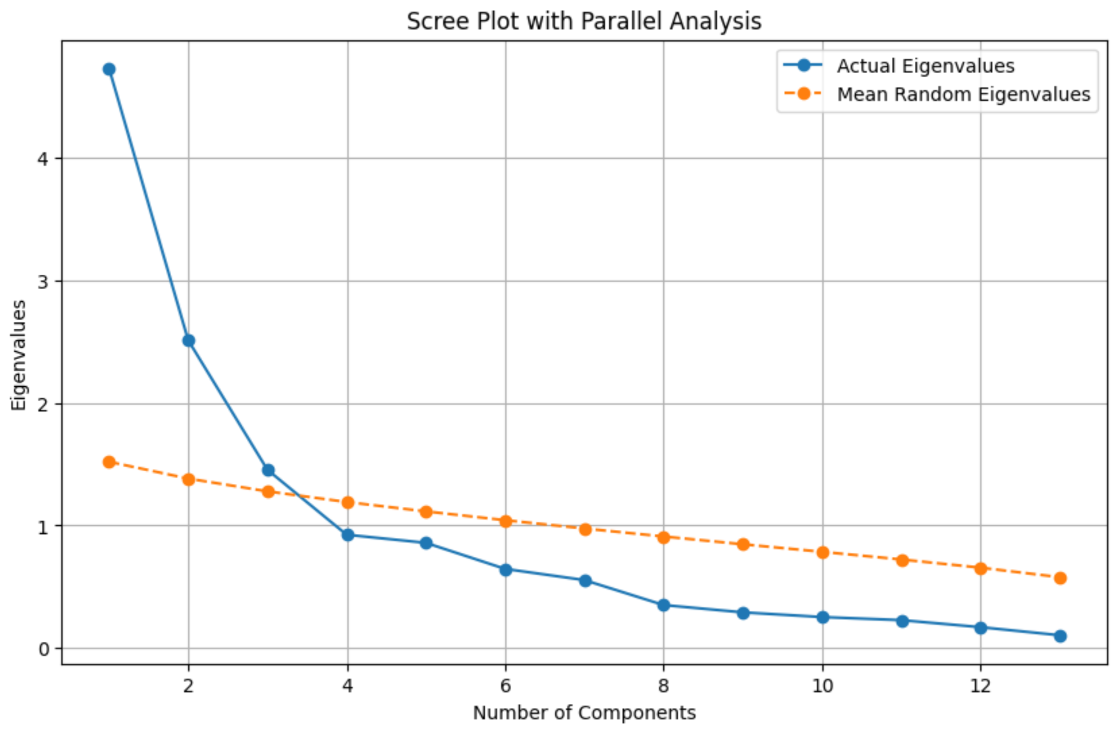
0.2.2 [2.2] Parallel Analysis

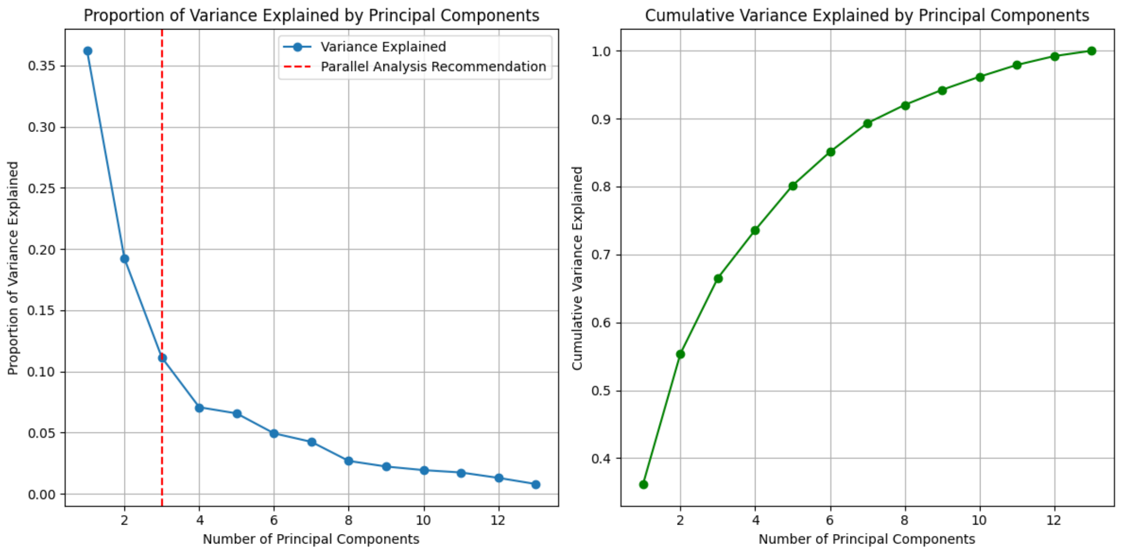
[65]:
```
def parallel_analysis(data, n_components):
    pca = PCA(n_components=n_components)
    pca.fit(data)
    eigenvalues = pca.explained_variance_
    num_iterations = 1000
    random_eigenvalues = np.zeros((num_iterations, len(eigenvalues)))
    for i in range(num_iterations):
        random_data = np.random.normal(0, 1, size=data.shape)
        pca.fit(random_data)
        random_eigenvalues[i, :] = pca.explained_variance_
    mean_random_eigenvalues = np.mean(random_eigenvalues, axis=0)
    diff_eigenvalues = eigenvalues - mean_random_eigenvalues
    plt.figure(figsize=(10, 6))
    components_range = np.arange(1, len(eigenvalues) + 1)
    plt.plot(components_range, eigenvalues, marker='o', label='Actual␣
↪Eigenvalues')
    plt.plot(components_range, mean_random_eigenvalues, marker='o',␣
↪linestyle='--', label='Mean Random Eigenvalues')
    plt.title('Scree Plot with Parallel Analysis')
    plt.xlabel('Number of Components')
    plt.ylabel('Eigenvalues')
    plt.legend()
    plt.grid(True)
    plt.show()
    return diff_eigenvalues

n_components = X.shape[1]
diff_eigenvalues = parallel_analysis(X_scaled, n_components)
num_components_pa = np.sum(diff_eigenvalues > 0)
pca = PCA()
pca.fit(X_scaled)
variance_explained = pca.explained_variance_ratio_
cumulative_variance_explained = np.cumsum(variance_explained)

plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.plot(range(1, len(variance_explained) + 1), variance_explained, marker='o',␣
↪label='Variance Explained')
plt.axvline(x=num_components_pa, color='r', linestyle='--', label='Parallel␣
↪Analysis Recommendation')
plt.title('Proportion of Variance Explained by Principal Components')
plt.xlabel('Number of Principal Components')
plt.ylabel('Proportion of Variance Explained')
plt.legend()
plt.grid(True)
plt.subplot(1, 2, 2)
plt.plot(range(1, len(cumulative_variance_explained) + 1),␣
↪cumulative_variance_explained, marker='o', color='g')
plt.title('Cumulative Variance Explained by Principal Components')
plt.xlabel('Number of Principal Components')
plt.ylabel('Cumulative Variance Explained')
plt.grid(True)
plt.tight_layout()
plt.show()

print("Number of principal components to retain based on parallel analysis:",␣
↪num_components_pa)
print("Proportion of variance explained by each principal component:")
for i, var in enumerate(variance_explained):
    print(f"Principal Component {i+1}: {var:.2f}")
if num_components_pa > len(variance_explained):
    print("Parallel analysis recommends retaining more components than the␣
↪proportion of variance explained criterion.")
elif num_components_pa < len(variance_explained):
    print("Parallel analysis recommends retaining fewer components than the␣
↪proportion of variance explained criterion.")
else:
    print("Parallel analysis and proportion of variance explained criterion␣
↪recommend retaining the same number of components.")
```





```
Number of principal components to retain based on parallel analysis: 3
Proportion of variance explained by each principal component:
Principal Component 1: 0.36
Principal Component 2: 0.19
Principal Component 3: 0.11
Principal Component 4: 0.07
Principal Component 5: 0.07
Principal Component 6: 0.05
Principal Component 7: 0.04
Principal Component 8: 0.03
Principal Component 9: 0.02
Principal Component 10: 0.02
Principal Component 11: 0.02
Principal Component 12: 0.01
Principal Component 13: 0.01
Parallel analysis recommends retaining fewer components than the proportion of
variance explained criterion.
```

0.2.3 [2.3] Proportion of Variance

[66]:
```
pca = PCA(n_components=3)
pca.fit(df)
variance_explained = pca.explained_variance_ratio_
print("Proportion of variance explained by the first three principal components:
 ↪")
for i, explained_variance in enumerate(variance_explained):
    print(f"Principal Component {i+1}: {explained_variance:.2f}")
total_variance_explained = sum(variance_explained)
```

```
print(f"Total variance explained by the first three principal components:␣
↪{total_variance_explained:.2f}")
```

```
Proportion of variance explained by the first three principal components:
Principal Component 1: 0.36
Principal Component 2: 0.19
Principal Component 3: 0.11
Total variance explained by the first three principal components: 0.67
```

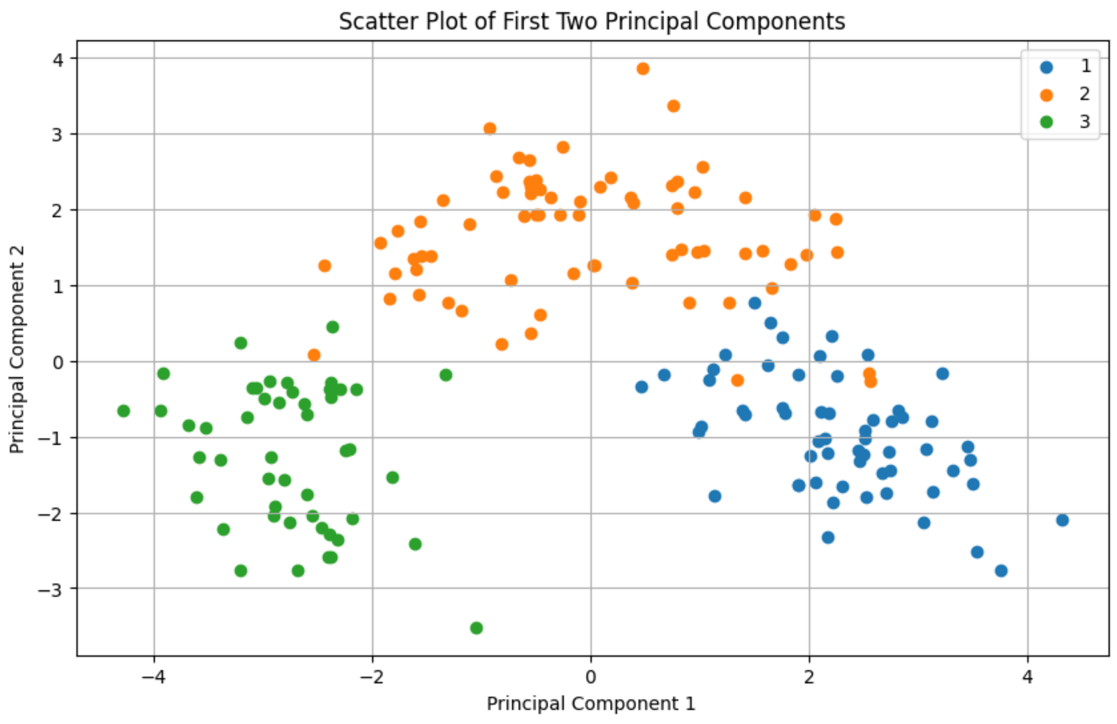
Parallel Analysis suggests retaining 3 principal components.

Proportion of Variance Explained Criterion indicates that the first 3 principal components collectively account for 66.52997% of the dataset's information, indicating sufficiency.

```
Comparison: Both methods concur on retaining 3 principal components.
```

0.2.4 [2.4] Scatter plots for the first two principal components

[67]:
```
pca_transformed = pca.transform(df)
pc1 = pca_transformed[:, 0]
pc2 = pca_transformed[:, 1]
plt.figure(figsize=(10, 6))
for wine_type in y.unique():
    plt.scatter(pc1[y == wine_type],
                pc2[y == wine_type],
                label=wine_type)
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.title('Scatter Plot of First Two Principal Components')
plt.legend()
plt.grid(True)
plt.show()
```



Interpretation: Significant overlap exists between wine types 1 and 2, with a lesser overlap between wine types 2 and 3.

These overlaps imply that the first two principal components might not sufficiently distinguish between the wine types.

It suggests that factors beyond these components could influence the dataset's variability and the differentiation of wine types.

To enhance the separation among wine types, consideration of additional principal components could be beneficial.

0.3 [3] Chi-Square Test - Goodness of Fit

0.3.1 [3.1] Hypotheses:

Null Hypothesis (H0): The observed distribution of wine types does not significantly differ from an equal distribution amongst the three types.

Alternative Hypothesis (H1): The observed distribution of wine types significantly differs from an equal distribution amongst the three types.

0.3.2 Expected Frequencies:

Since we assume an equal distribution among the three types, each type is expected to have 178/3 = 59.33 occurrences.

0.3.3 [3.2] Performing Chi-Square Goodness of Fit Test

[68]:
```
observed_frequencies = np.array([59, 71, 48])
expected_frequency = 178 / 3
expected_frequencies = np.array([expected_frequency, expected_frequency,␣
 ↪expected_frequency])
chi2_statistic, p_value = chisquare(observed_frequencies,␣
 ↪f_exp=expected_frequencies)
df = len(observed_frequencies) - 1
print("Chi-Square Statistic:", chi2_statistic)
print("Degrees of Freedom:", df)
print("p-value:", p_value)
```

```
Chi-Square Statistic: 4.4606741573033695
Degrees of Freedom: 2
p-value: 0.10749219070565924
```

0.3.4 [3.3] Interpretation

- If the p-value is below the significance level, we reject the null hypothesis, indicating significant evidence of deviation from an equal wine type distribution.
- When the p-value exceeds the significance level, we fail to reject the null hypothesis, suggesting insufficient evidence to conclude a significant deviation from equal distribution.

[69]:
```
alpha = 0.05
if p_value <= alpha:
    print("Reject the null hypothesis. The observed distribution significantly␣
 ↪differs from an equal distribution.")
else:
    print("Fail to reject the null hypothesis. The observed distribution does␣
 ↪not significantly differ from an equal distribution.")
```

```
Fail to reject the null hypothesis. The observed distribution does not
significantly differ from an equal distribution.
```

0.4 [4] Chi-Square Test - Independence

0.4.1 [4.1] Categorise Alcohol Content

[70]:
```
low_percentile = original_data['Alcohol'].quantile(0.33)
high_percentile = original_data['Alcohol'].quantile(0.66)
def categorize_alcohol(alcohol):
    if alcohol <= low_percentile:
        return 'Low'
    elif low_percentile < alcohol <= high_percentile:
        return 'Medium'
    else:
        return 'High'
```

0.4.2 [4.2] Contingency Table

[71]:
```
original_data['Alcohol_Category'] = original_data['Alcohol'].
 ↪apply(categorize_alcohol)
contingency_table = pd.crosstab(original_data['Type'],␣
 ↪original_data['Alcohol_Category'])
```

0.4.3 [4.3] Hypotheses

- Null hypothesis (H0): The choice of wine type is independent of the alcohol content.
- Alternative hypothesis (Ha): The choice of wine type is dependent on the alcohol content.

0.4.4 [4.4] Performing Chi-Squared Test

[72]:
```
chi2_statistic, p_value, dof, _ = chi2_contingency(contingency_table)
alpha = 0.05
print("Contingency Table (Cross-tabulation):")
print(contingency_table)
print("\nChi-Squared Test Results:")
print(f"Chi-Square Statistic: {chi2_statistic}")
print(f"P-value: {p_value}")
print(f"Degrees of Freedom: {dof}")
```

```
Contingency Table (Cross-tabulation):
Alcohol_Category  High  Low  Medium
Type
1                   43    0      16
2                    3   53      15
3                   15    6      27

Chi-Squared Test Results:
Chi-Square Statistic: 118.77146680058311
P-value: 9.77322789202807e-25
Degrees of Freedom: 4
```

0.4.5 [4.5] Interpretation

- If the p-value is lower than the significance level, we reject the null hypothesis, indicating significant evidence of an association between wine type preference and alcohol content level (Low, Medium, High).
- If the p-value exceeds the significance level, we fail to reject the null hypothesis, signifying that we cannot conclude a significant association between wine type preference and alcohol

content level (Low, Medium, High) based on our data.

[73]:
```
if p_value <= alpha:
    print("Reject the null hypothesis. There is an association between wine␣
 ↪type and alcohol category.")
else:
    print("Fail to reject the null hypothesis. There is no association between␣
 ↪wine type and alcohol category.")
```

```
Reject the null hypothesis. There is an association between wine type and
alcohol category.
```