Unsupervised Learning for Wine Classification: A Chemical Composition Analysis

Ahsan Khan

Unsupervised Algorithms in Machine Learning

University of Colorado Boulder

Dataset Overview

- We’re working with “Wine” dataset from UCI Machine Learning repository.
- Dataset contains results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars
- Number of Samples: 178 wines
- Number of Features: 13 chemical attributes
- Feature Types: All features are continuous numerical values

Project Objective

Analyze and classify wines based on their chemical attributes

Identify key chemical components that significantly contribute to wine differentiation

Discover natural clusters within the wine dataset that could represent distinct wine types

Compare and evaluate different clustering algorithms to determine the most effective approach for wine classification

Provide insights that could assist the wine industry in quality control, product development, and marketing strategies

Methodology

- Data Preprocessing
- Exploratory Data Analysis (EDA)
- Dimensionality Reduction using Principal Component Analysis (PCA)
- Feature Engineering
- Clustering Analysis using Multiple Algorithms
- Comparison and Evaluation of Clustering Results
- Conclusion

Load and Inspect the Data

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
df = pd.read_csv('wine-clustering.csv')
```

```python
# Display the first few rows and basic information about the dataset
print(df.head())
print("\nDataset Information:")
print(df.info())
```

```
   Alcohol  Malic_Acid   Ash  Ash_Alcanity  Magnesium  Total_Phenols  \
0    14.23        1.71  2.43          15.6        127           2.80
1    13.20        1.78  2.14          11.2        100           2.65
2    13.16        2.36  2.67          18.6        101           2.80
3    14.37        1.95  2.50          16.8        113           3.85
4    13.24        2.59  2.87          21.0        118           2.80

   Flavanoids  Nonflavanoid_Phenols  Proanthocyanins  Color_Intensity   Hue  \
0        3.06                  0.28             2.29             5.64  1.04
1        2.76                  0.26             1.28             4.38  1.05
2        3.24                  0.30             2.81             5.68  1.03
3        3.49                  0.24             2.18             7.80  0.86
4        2.69                  0.39             1.82             4.32  1.04

   OD280  Proline
0   3.92     1065
1   3.40     1050
2   3.17     1185
3   3.45     1480
4   2.93      735

Dataset Information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 178 entries, 0 to 177
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Alcohol               178 non-null    float64
 1   Malic_Acid            178 non-null    float64
 2   Ash                   178 non-null    float64
 3   Ash_Alcanity          178 non-null    float64
 4   Magnesium             178 non-null    int64
 5   Total_Phenols         178 non-null    float64
 6   Flavanoids            178 non-null    float64
 7   Nonflavanoid_Phenols  178 non-null    float64
 8   Proanthocyanins       178 non-null    float64
 9   Color_Intensity       178 non-null    float64
 10  Hue                   178 non-null    float64
 11  OD280                 178 non-null    float64
 12  Proline               178 non-null    int64
dtypes: float64(11), int64(2)
memory usage: 18.2 KB
None
```

Data Cleaning

```
# Check for missing values
print("\nMissing Values:")
print(df.isnull().sum())
```

```
Missing Values:
Alcohol                 0
Malic_Acid              0
Ash                     0
Ash_Alcanity            0
Magnesium               0
Total_Phenols           0
Flavanoids              0
Nonflavanoid_Phenols    0
Proanthocyanins         0
Color_Intensity         0
Hue                     0
OD280                   0
Proline                 0
dtype: int64
```

Basic Statistics

```
# Display basic statistics of the dataset
print("\nBasic Statistics:")
print(df.describe())
```

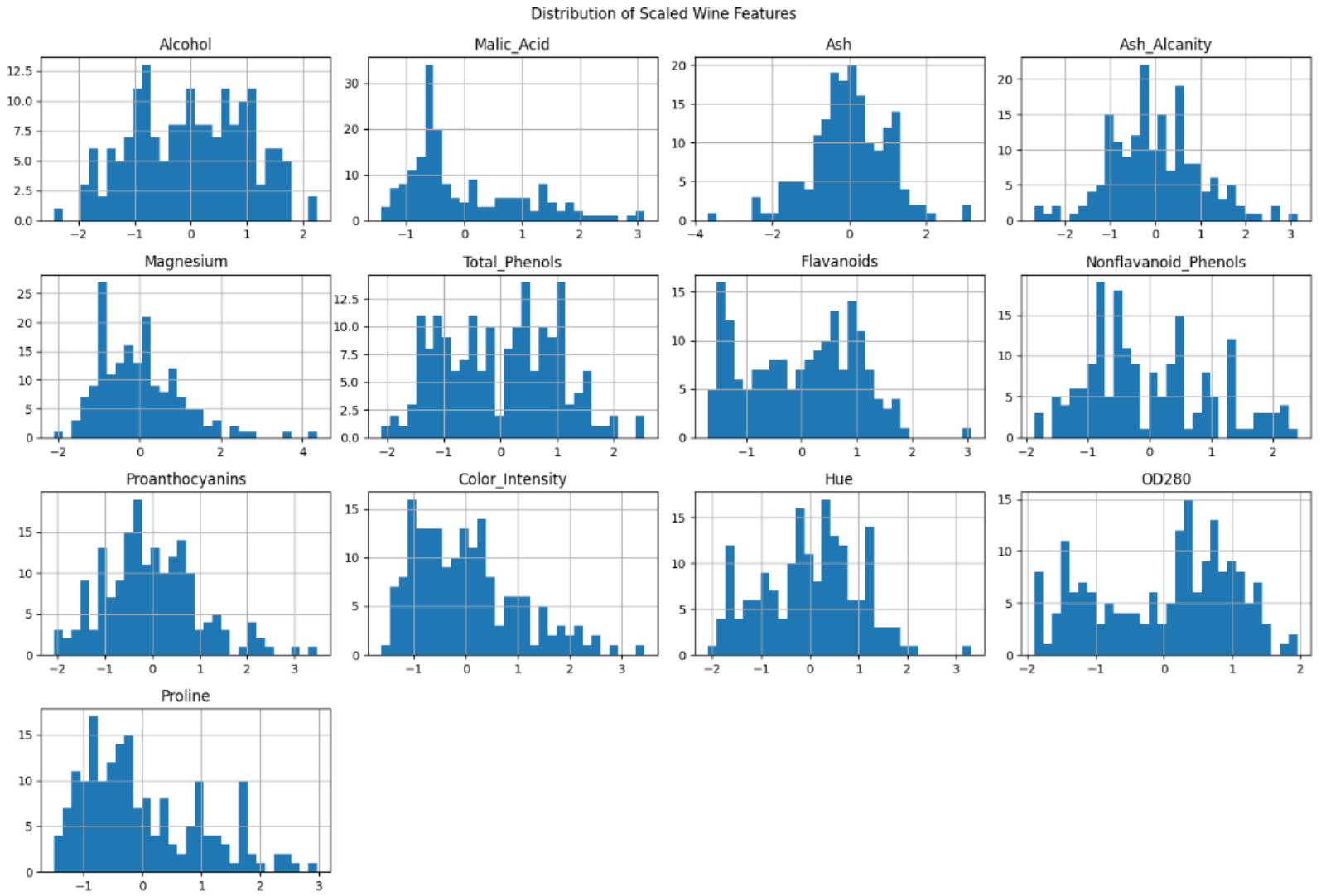
```
Basic Statistics:
          Alcohol  Malic_Acid         Ash  Ash_Alcanity   Magnesium  \
count  178.000000  178.000000  178.000000    178.000000  178.000000
mean    13.000618    2.336348    2.366517     19.494944   99.741573
std      0.811827    1.117146    0.274344      3.339564   14.282484
min     11.030000    0.740000    1.360000     10.600000   70.000000
25%     12.362500    1.602500    2.210000     17.200000   88.000000
50%     13.050000    1.865000    2.360000     19.500000   98.000000
75%     13.677500    3.082500    2.557500     21.500000  107.000000
max     14.830000    5.800000    3.230000     30.000000  162.000000

       Total_Phenols  Flavanoids  Nonflavanoid_Phenols  Proanthocyanins  \
count     178.000000  178.000000            178.000000       178.000000
mean        2.295112    2.029270              0.361854         1.590899
std         0.625851    0.998859              0.124453         0.572359
min         0.980000    0.340000              0.130000         0.410000
25%         1.742500    1.205000              0.270000         1.250000
50%         2.355000    2.135000              0.340000         1.555000
75%         2.800000    2.875000              0.437500         1.950000
max         3.880000    5.080000              0.660000         3.580000

       Color_Intensity         Hue       OD280      Proline
count       178.000000  178.000000  178.000000   178.000000
mean          5.058090    0.957449    2.611685   746.893258
std           2.318286    0.228572    0.709990   314.907474
min           1.280000    0.480000    1.270000   278.000000
25%           3.220000    0.782500    1.937500   500.500000
50%           4.690000    0.965000    2.780000   673.500000
75%           6.200000    1.120000    3.170000   985.000000
max          13.000000    1.710000    4.000000  1680.000000
```

Feature Scaling

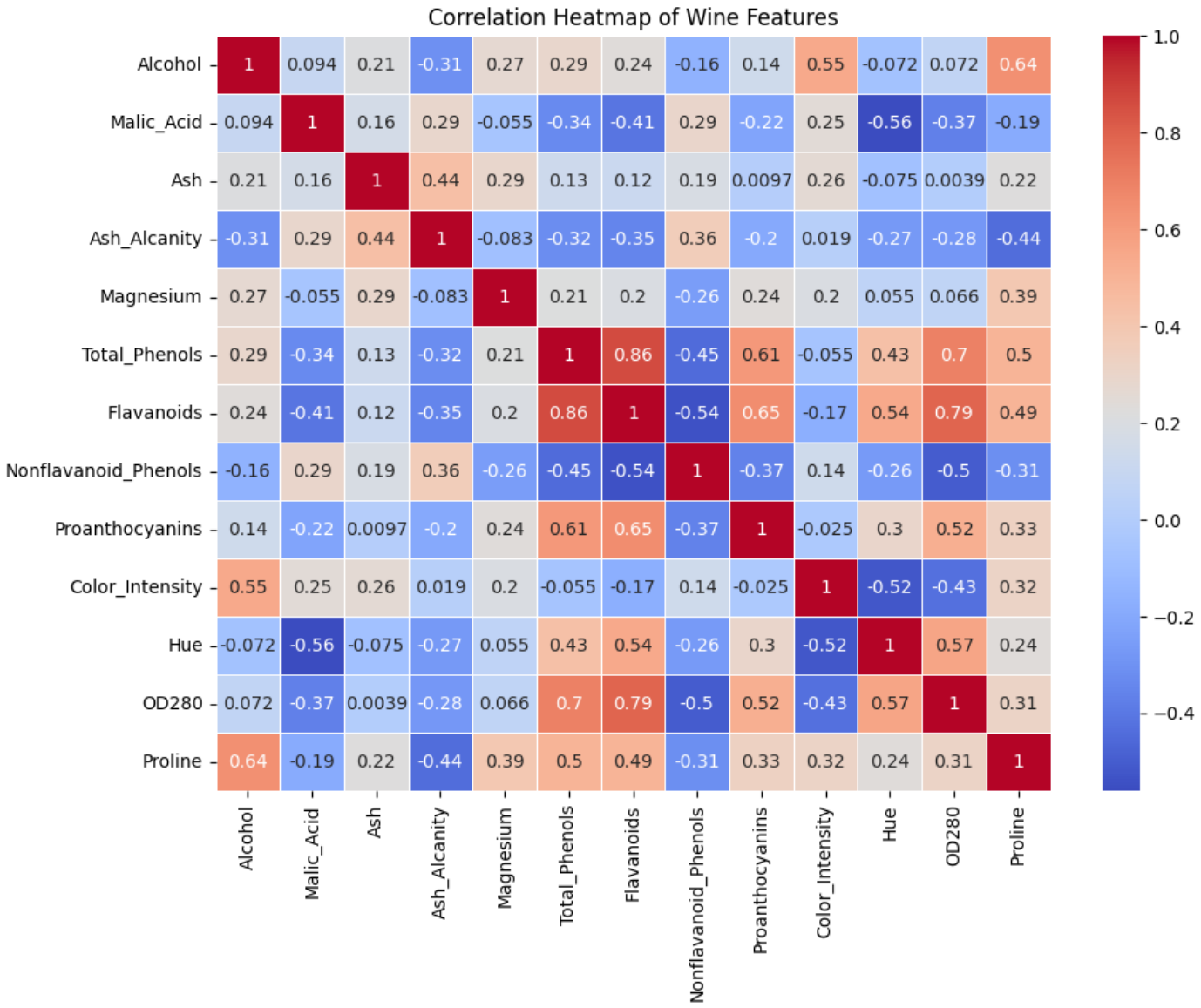
```python
from sklearn.preprocessing import StandardScaler

# Initialize the StandardScaler
scaler = StandardScaler()

# Fit and transform the data
df_scaled = pd.DataFrame(scaler.fit_transform(df), columns=df.columns)

# Display basic statistics of the scaled dataset
print(df_scaled.describe())
```

```
            Alcohol    Malic_Acid           Ash  Ash_Alcanity     Magnesium  \
count  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
std    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
25%   -7.882448e-01 -6.587486e-01 -5.721225e-01 -6.891372e-01 -8.244151e-01
50%    6.099988e-02 -4.231120e-01 -2.382132e-02  1.518295e-03 -1.222817e-01
75%    8.361286e-01  6.697929e-01  6.981085e-01  6.020883e-01  5.096384e-01
max    2.259772e+00  3.109192e+00  3.156325e+00  3.154511e+00  4.371372e+00

       Total_Phenols    Flavanoids  Nonflavanoid_Phenols  Proanthocyanins  \
count     178.000000  1.780000e+02          1.780000e+02     1.780000e+02
mean        0.000000 -3.991813e-16          3.592632e-16    -1.197544e-16
std         1.002821  1.002821e+00          1.002821e+00     1.002821e+00
min        -2.107246 -1.695971e+00         -1.868234e+00    -2.069034e+00
25%        -0.885468 -8.275393e-01         -7.401412e-01    -5.972835e-01
50%         0.095960  1.061497e-01         -1.760948e-01    -6.289785e-02
75%         0.808997  8.490851e-01          6.095413e-01     6.291754e-01
max         2.539515  3.062832e+00          2.402403e+00     3.485073e+00

       Color_Intensity           Hue         OD280       Proline
count     1.780000e+02  1.780000e+02  1.780000e+02  1.780000e+02
mean      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
min      -1.634288e+00 -2.094732e+00 -1.895054e+00 -1.493188e+00
25%      -7.951025e-01 -7.675624e-01 -9.522483e-01 -7.846378e-01
50%      -1.592246e-01  3.312687e-02  2.377348e-01 -2.337204e-01
75%       4.939560e-01  7.131644e-01  7.885875e-01  7.582494e-01
max       3.435432e+00  3.301694e+00  1.960915e+00  2.971473e+00
```

Exploratory Data Analysis (EDA)



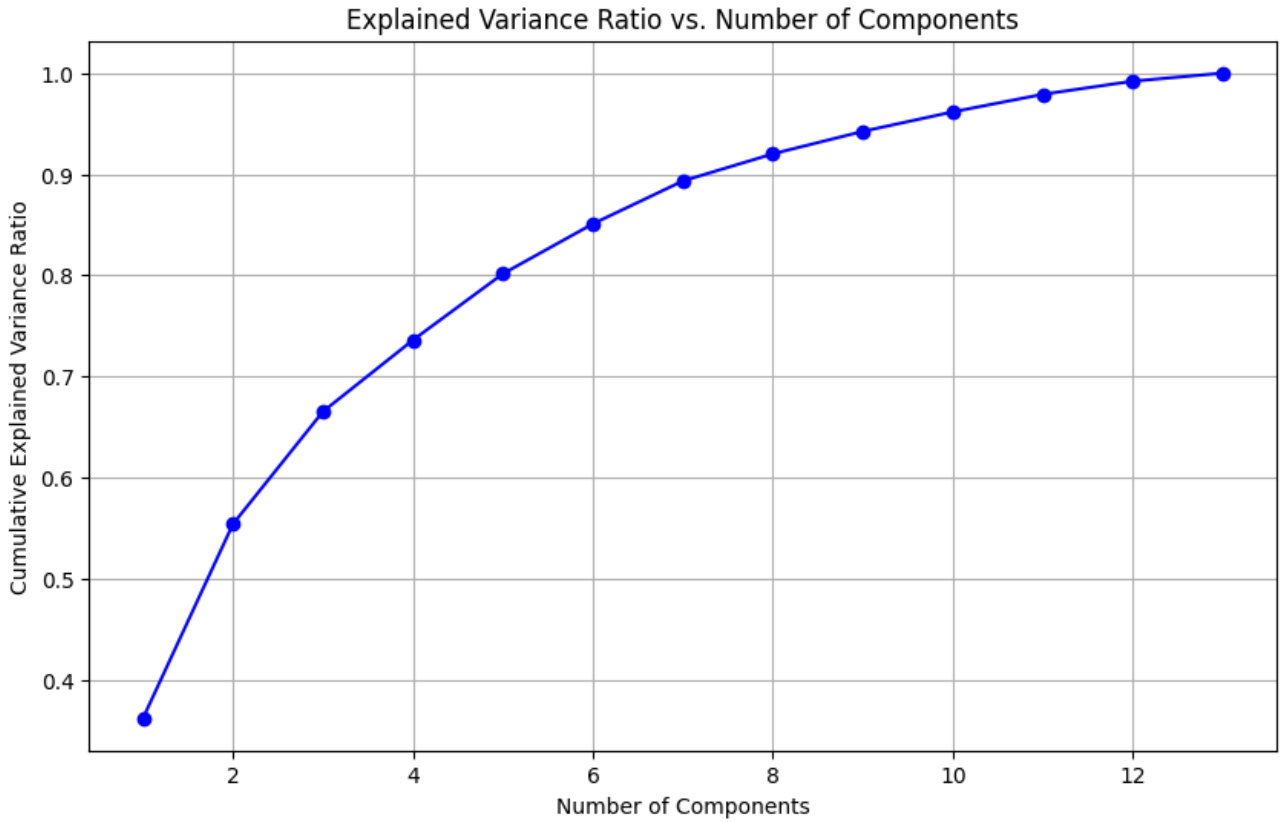
Correlation Heatmap



Dimensionality Reduction with PCA

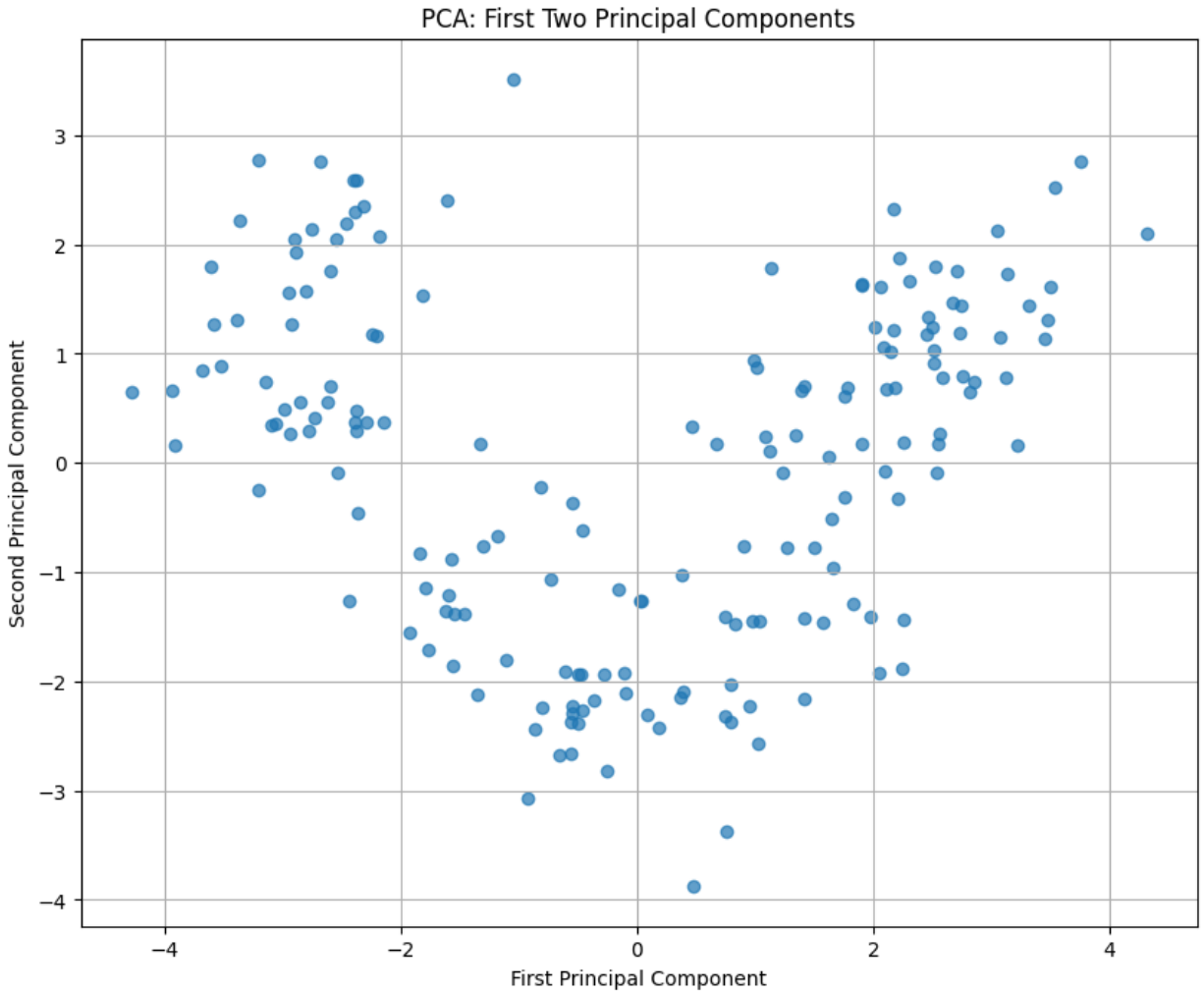
```python
from sklearn.decomposition import PCA
import numpy as np

# Perform PCA
pca = PCA()
pca_result = pca.fit_transform(df_scaled)

# Calculate cumulative explained variance ratio
cumulative_variance_ratio = np.cumsum(pca.explained_variance_ratio_)

# Plot cumulative explained variance ratio
plt.figure(figsize=(10, 6))
plt.plot(range(1, len(cumulative_variance_ratio) + 1), cumulative_variance_ratio, 'bo-')
plt.xlabel('Number of Components')
plt.ylabel('Cumulative Explained Variance Ratio')
plt.title('Explained Variance Ratio vs. Number of Components')
plt.grid(True)
plt.show()

# Print explained variance ratio for each component
for i, ratio in enumerate(pca.explained_variance_ratio_):
    print(f"PC{i+1} Explained Variance Ratio: {ratio:.4f}")

# Create a scatter plot of the first two principal components
plt.figure(figsize=(10, 8))
plt.scatter(pca_result[:, 0], pca_result[:, 1], alpha=0.7)
plt.xlabel('First Principal Component')
plt.ylabel('Second Principal Component')
plt.title('PCA: First Two Principal Components')
plt.grid(True)
plt.show()

# Create a DataFrame with PCA results for further analysis
pca_df = pd.DataFrame(data=pca_result[:, :2], columns=['PC1', 'PC2'])
print(pca_df.head())
```

Explained Variance Ratio



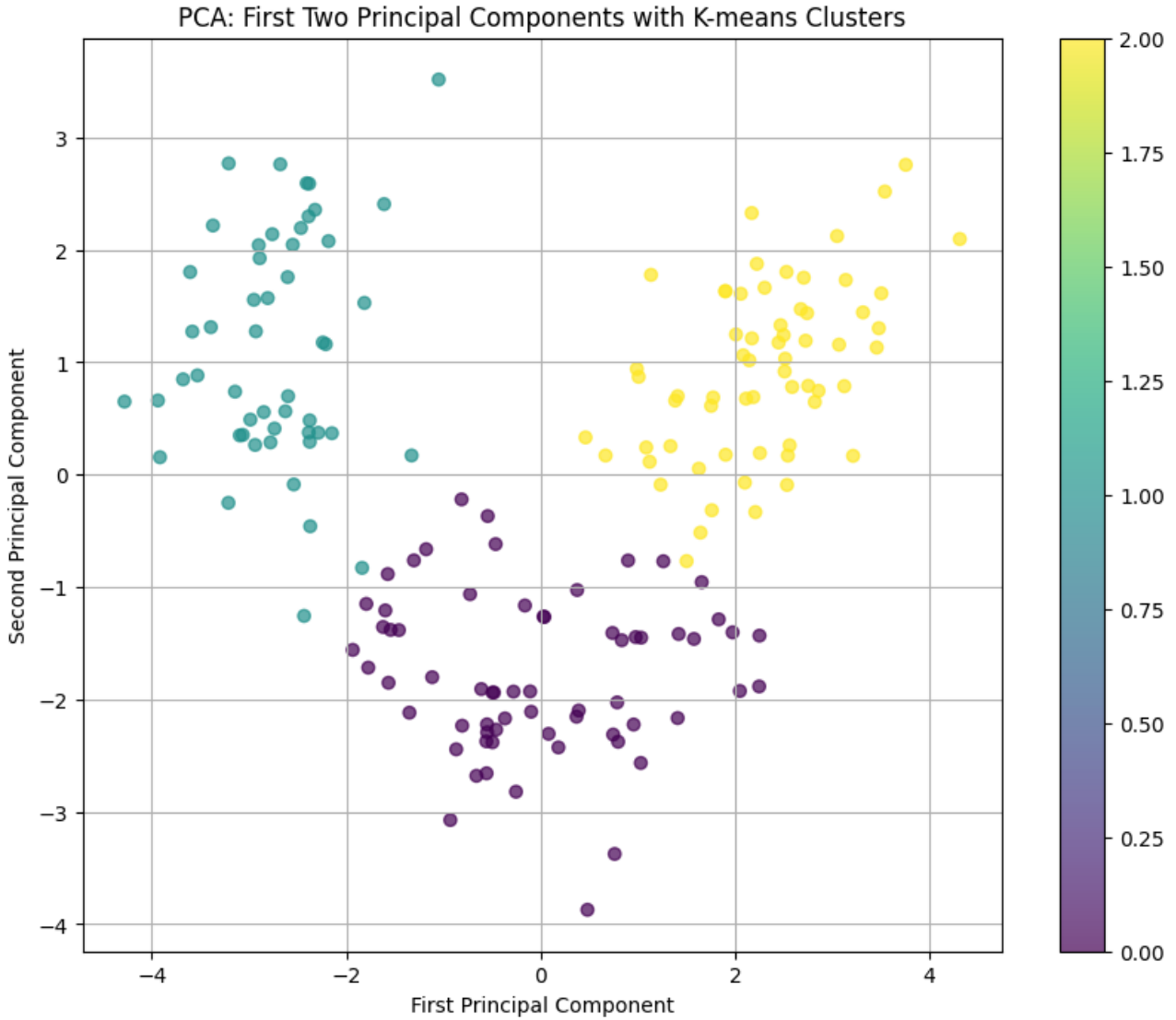

```
PC1 Explained Variance Ratio: 0.3620
PC2 Explained Variance Ratio: 0.1921
PC3 Explained Variance Ratio: 0.1112
PC4 Explained Variance Ratio: 0.0707
PC5 Explained Variance Ratio: 0.0656
PC6 Explained Variance Ratio: 0.0494
PC7 Explained Variance Ratio: 0.0424
PC8 Explained Variance Ratio: 0.0268
PC9 Explained Variance Ratio: 0.0222
PC10 Explained Variance Ratio: 0.0193
PC11 Explained Variance Ratio: 0.0174
PC12 Explained Variance Ratio: 0.0130
PC13 Explained Variance Ratio: 0.0080
```

First Two Principal Components



	PC1	PC2
0	3.316751	1.443463
1	2.209465	-0.333393
2	2.516740	1.031151
3	3.757066	2.756372
4	1.008908	0.869831

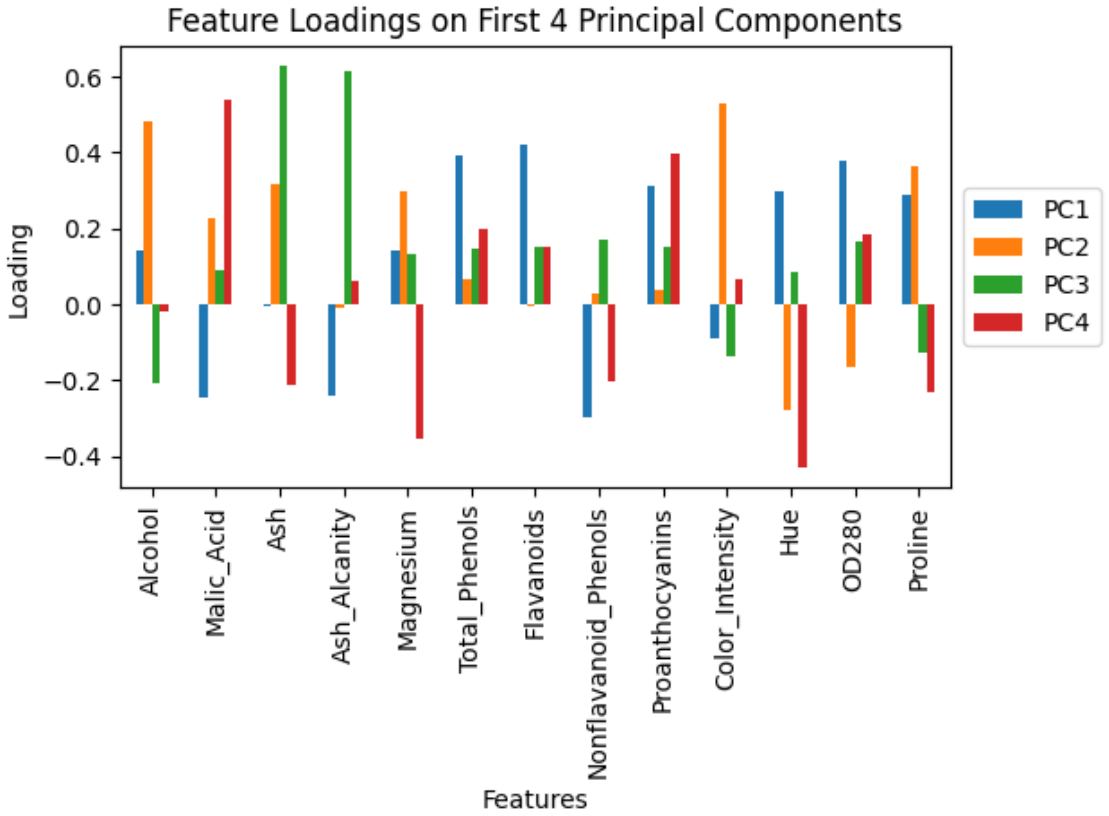
K-means Clustering & Further PCA Analysis



The average silhouette score is: 0.4051

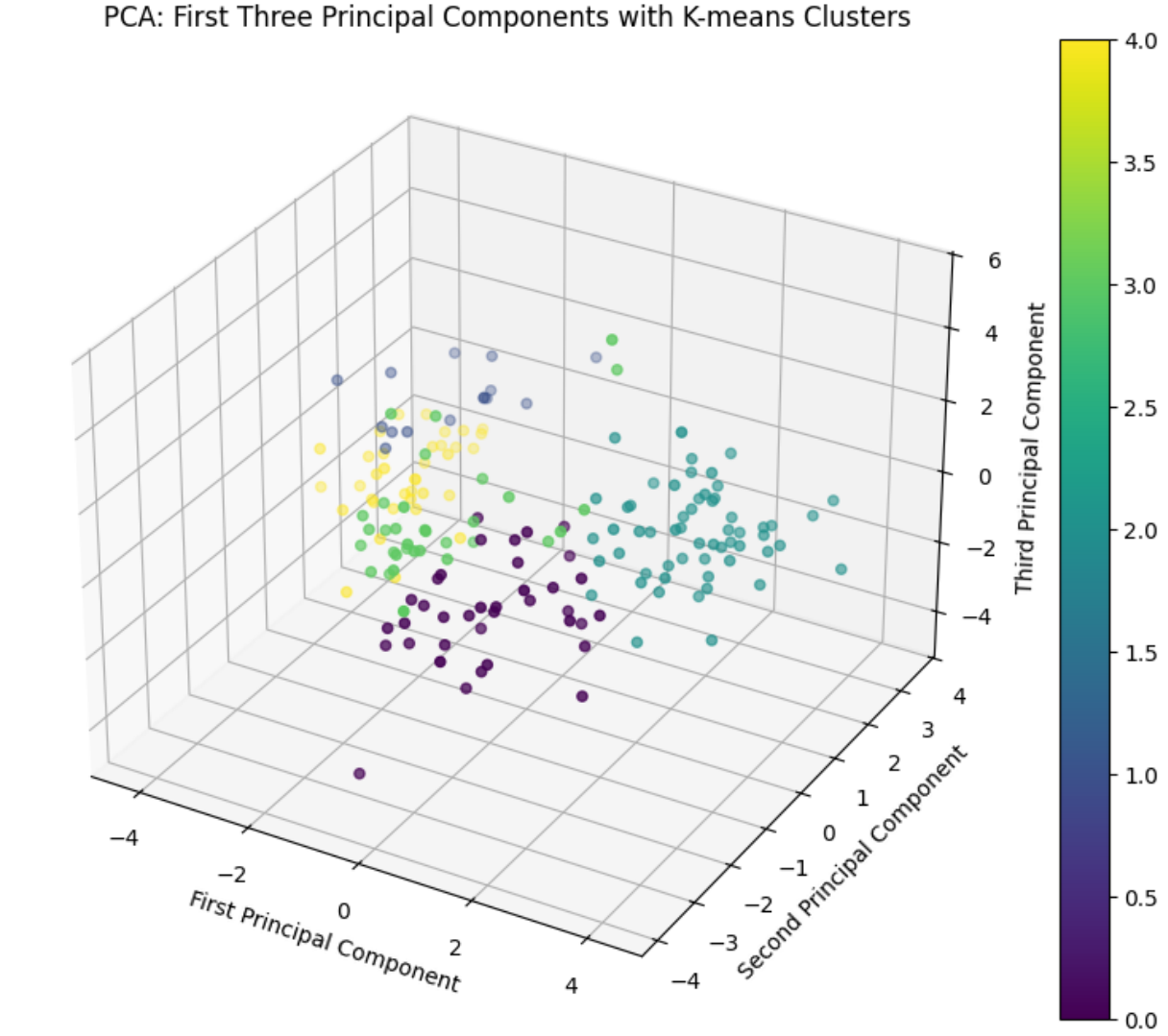
Feature Loadings

```
Feature loadings on first 4 PCs:
                              PC1       PC2       PC3       PC4
Alcohol                  0.144329  0.483652 -0.207383 -0.017856
Malic_Acid              -0.245188  0.224931  0.089013  0.536890
Ash                     -0.002051  0.316069  0.626224 -0.214176
Ash_Alcanity            -0.239320 -0.010591  0.612080  0.060859
Magnesium                0.141992  0.299634  0.130757 -0.351797
Total_Phenols            0.394661  0.065040  0.146179  0.198068
Flavanoids               0.422934 -0.003360  0.150682  0.152295
Nonflavanoid_Phenols    -0.298533  0.028779  0.170368 -0.203301
Proanthocyanins          0.313429  0.039302  0.149454  0.399057
Color_Intensity         -0.088617  0.529996 -0.137306  0.065926
Hue                      0.296715 -0.279235  0.085222 -0.427771
OD280                    0.376167 -0.164496  0.166005  0.184121
Proline                  0.286752  0.364903 -0.126746 -0.232071
<Figure size 1200x800 with 0 Axes>
```

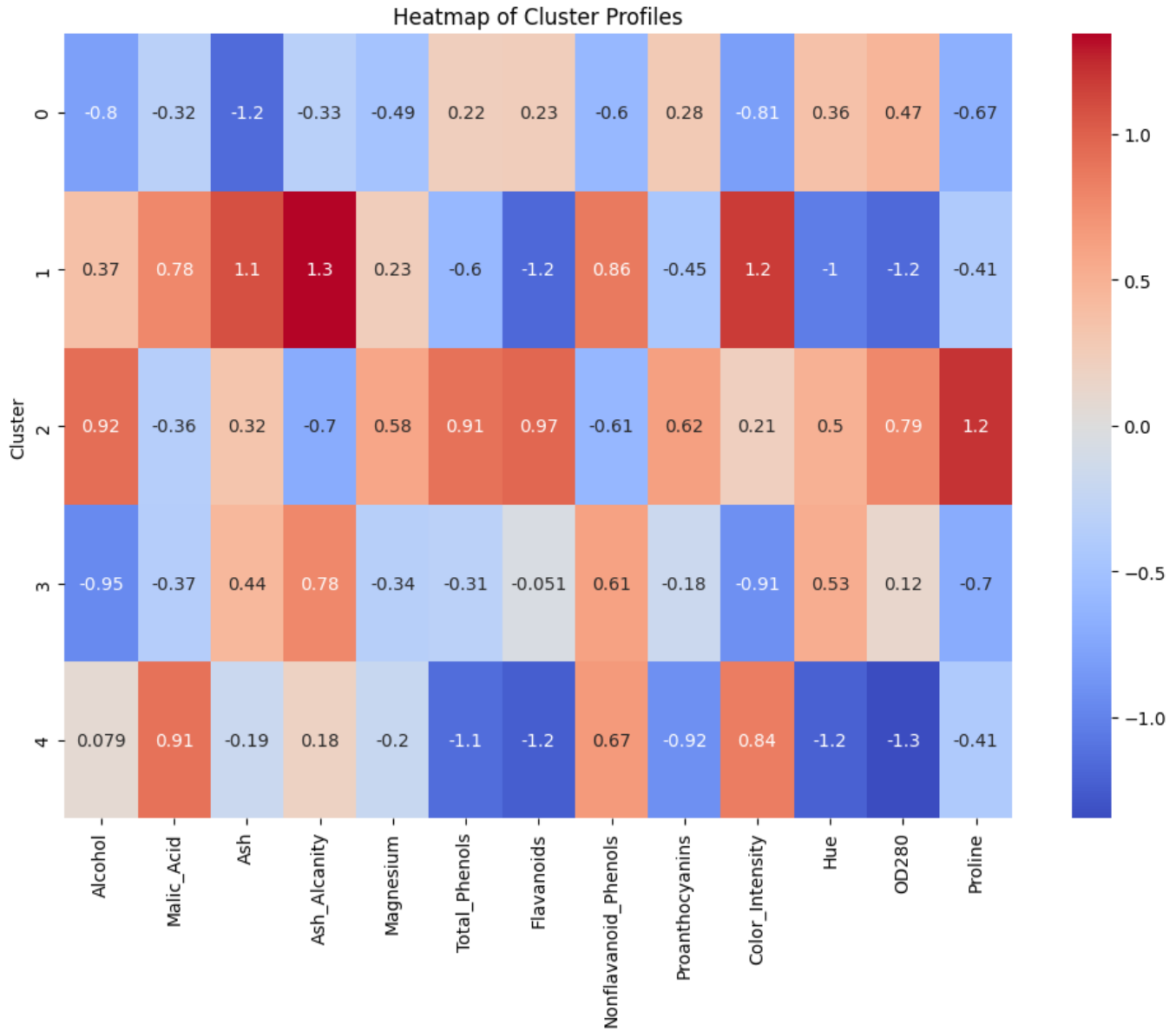


Experiment with Different Numbers of Clusters

```
For n_clusters = 2, the average silhouette score is: 0.3531
For n_clusters = 3, the average silhouette score is: 0.4051
For n_clusters = 4, the average silhouette score is: 0.3654
For n_clusters = 5, the average silhouette score is: 0.3009
```

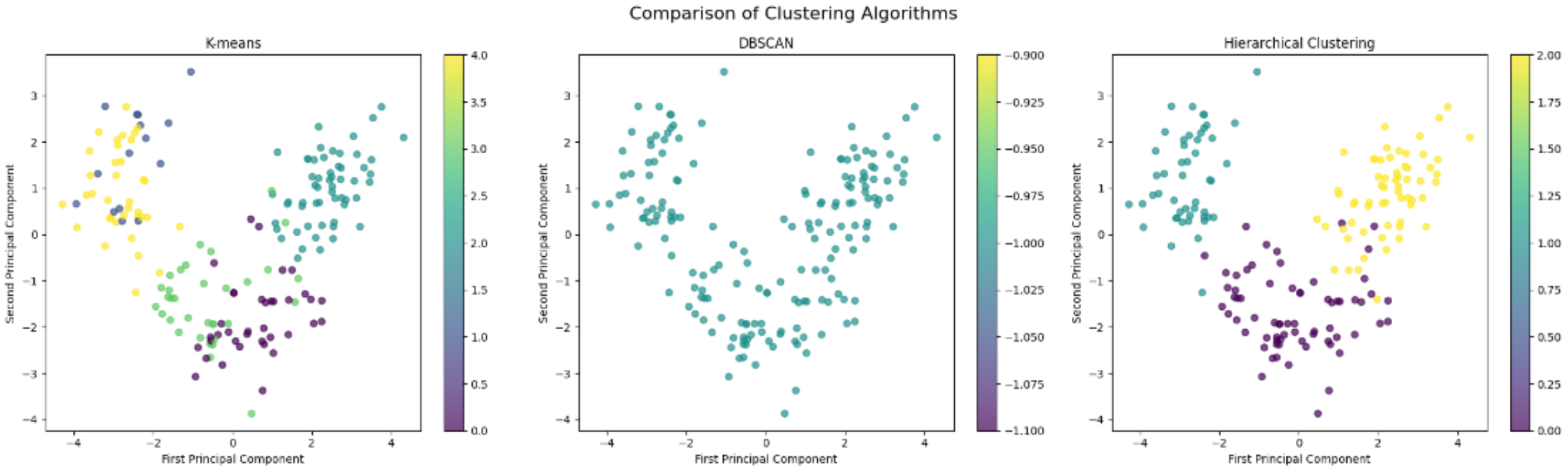


Heatmap of Cluster Profiles



Comparison of Clustering Algorithms

```
K-means Silhouette Score: 0.3009
DBSCAN did not identify multiple clusters.
Hierarchical Clustering Silhouette Score: 0.3865
```



Cluster Comparisons

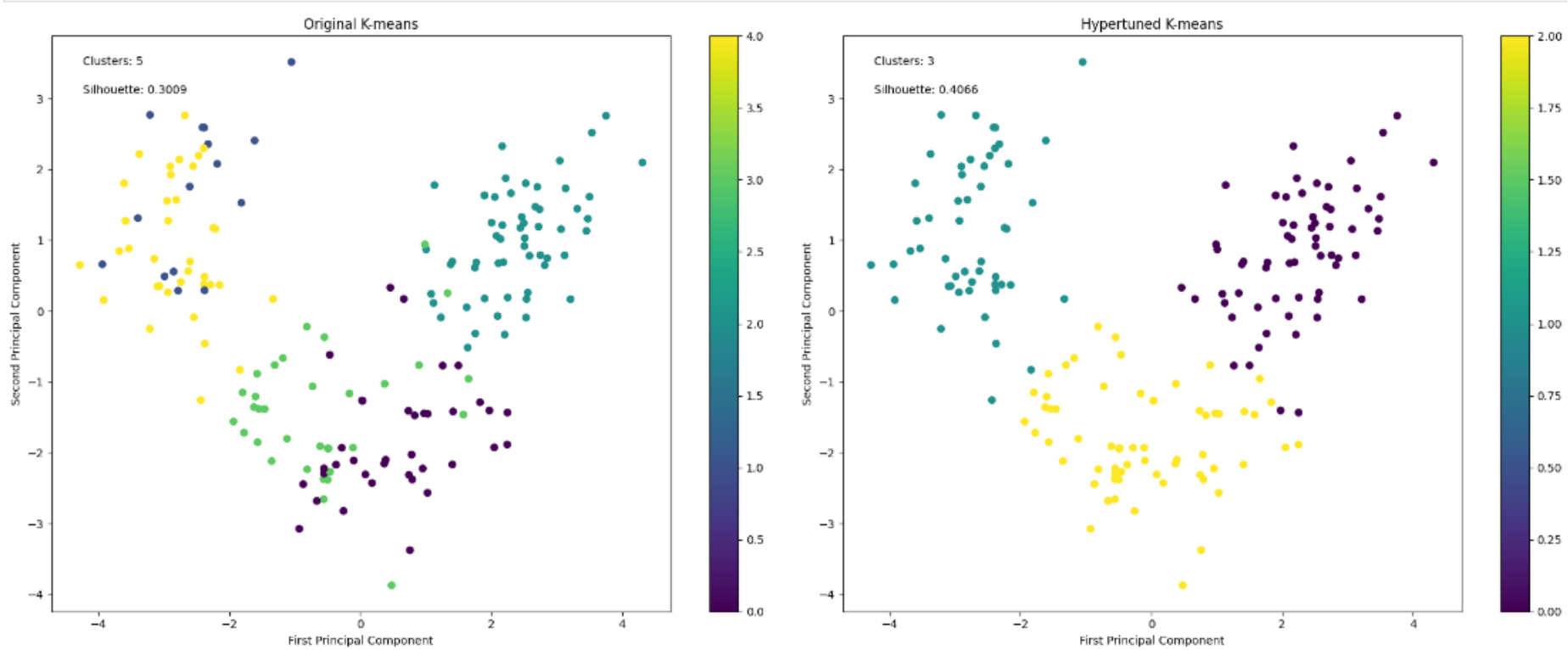
```
Cluster Comparisons:

Contingency Table (K-means vs DBSCAN):
col_0  -1
row_0
0      37
1      15
2      57
3      33
4      36

Contingency Table (K-means vs Hierarchical):
col_0   0   1   2
row_0
0      32   0   5
1       0  15   0
2       3   0  54
3      30   0   3
4       3  33   0

Contingency Table (DBSCAN vs Hierarchical):
col_0   0   1   2
row_0
-1     68  48  62
```

Hyperparameter Tuning

```
print("Inertia:", best_kmeans.inertia_)
```

```
Best parameters: {'init': 'k-means++', 'max_iter': 100, 'n_clusters': 3, 'n_init': 10}
Best silhouette score: -1.0
Cluster labels: [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 1 2 2 2 2 2 2 2 0 2 2 2 0
 2 2 2 2 0 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 0 2 2 0 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 1 2 2 0 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
Silhouette score for best model: 0.406596910894866
Inertia: 674.6349433774147
```

Original vs Hyper-tuned K-Means




```
Comparison Metrics:
Original K-means - Clusters: 5, Silhouette: 0.3009
Hypertuned K-means - Clusters: 3, Silhouette: 0.4066

Percentage of points with changed cluster assignments: 88.20%
```

Conclusion

- Effective dimensionality reduction using PCA
- Identification of key differentiating factors in wine composition
- Discovery of natural groupings in the wine data using multiple clustering approaches
- Comparative analysis of different clustering techniques
- Insights into the continuous nature of wine characteristics