```
##importing the dataset
import pandas as pd
df=pd.read_csv(r"C:\Users\USER\OneDrive\Data Science books\datasets\
Wine dataset\wine-clustering.csv")
df
```

```
     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

..       ...         ...   ...           ...        ...            ...

173    13.71        5.65  2.45          20.5         95           1.68

174    13.40        3.91  2.48          23.0        102           1.80

175    13.27        4.28  2.26          20.0        120           1.59

176    13.17        2.59  2.37          20.0        120           1.65

177    14.13        4.10  2.74          24.5         96           2.05


     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
..          ...                   ...              ...                ..
.   ...
173        0.61                  0.52             1.06
7.70  0.64
174        0.75                  0.43             1.41
7.30  0.70
175        0.69                  0.43             1.35
10.20  0.59
```

```
176        0.68                    0.53              1.46
9.30  0.60
177        0.76                    0.56              1.35
9.20  0.61

     OD280  Proline
0     3.92     1065
1     3.40     1050
2     3.17     1185
3     3.45     1480
4     2.93      735
..     ...      ...
173   1.74      740
174   1.56      750
175   1.56      835
176   1.62      840
177   1.60      560

[178 rows x 13 columns]
```

```
##data exploration
df.info()
```

```
<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
```

```
##summary statistics
df.describe()
```

```
          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
```

```
##correlation matrix
df.corr()
```

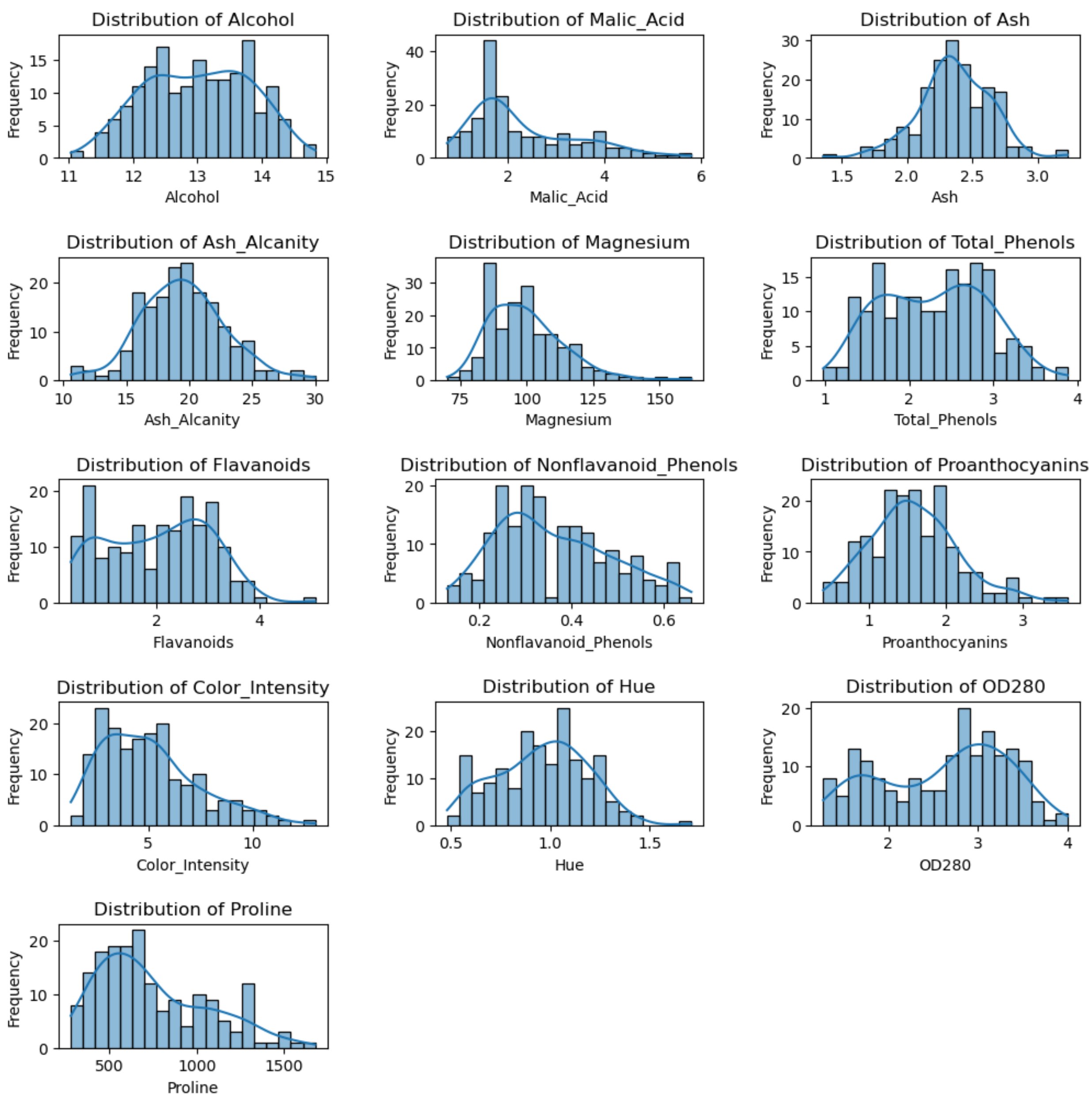
```
                       Alcohol  Malic_Acid       Ash  Ash_Alcanity
Magnesium  \
Alcohol               1.000000    0.094397  0.211545     -0.310235
0.270798
Malic_Acid            0.094397    1.000000  0.164045      0.288500  -
0.054575
Ash                   0.211545    0.164045  1.000000      0.443367
0.286587
Ash_Alcanity         -0.310235    0.288500  0.443367      1.000000  -
0.083333
Magnesium             0.270798   -0.054575  0.286587     -0.083333
```

```
1.000000
Total_Phenols         0.289101   -0.335167  0.128980     -0.321113
0.214401
Flavanoids            0.236815   -0.411007  0.115077     -0.351370
0.195784
Nonflavanoid_Phenols -0.155929    0.292977  0.186230      0.361922  -
0.256294
Proanthocyanins       0.136698   -0.220746  0.009652     -0.197327
0.236441
Color_Intensity       0.546364    0.248985  0.258887      0.018732
0.199950
Hue                  -0.071747   -0.561296 -0.074667     -0.273955
0.055398
OD280                 0.072343   -0.368710  0.003911     -0.276769
0.066004
Proline               0.643720   -0.192011  0.223626     -0.440597
0.393351

                      Total_Phenols  Flavanoids  Nonflavanoid_Phenols
\
Alcohol                    0.289101    0.236815             -0.155929

Malic_Acid                -0.335167   -0.411007              0.292977

Ash                        0.128980    0.115077              0.186230

Ash_Alcanity              -0.321113   -0.351370              0.361922

Magnesium                  0.214401    0.195784             -0.256294

Total_Phenols              1.000000    0.864564             -0.449935

Flavanoids                 0.864564    1.000000             -0.537900

Nonflavanoid_Phenols      -0.449935   -0.537900              1.000000

Proanthocyanins            0.612413    0.652692             -0.365845

Color_Intensity           -0.055136   -0.172379              0.139057

Hue                        0.433681    0.543479             -0.262640

OD280                      0.699949    0.787194             -0.503270

Proline                    0.498115    0.494193             -0.311385


                      Proanthocyanins  Color_Intensity       Hue
OD280  \
Alcohol                      0.136698         0.546364 -0.071747
0.072343
```

```
Malic_Acid                     -0.220746         0.248985 -0.561296 -
0.368710
Ash                             0.009652         0.258887 -0.074667
0.003911
Ash_Alcanity                   -0.197327         0.018732 -0.273955 -
0.276769
Magnesium                       0.236441         0.199950  0.055398
0.066004
Total_Phenols                   0.612413        -0.055136  0.433681
0.699949
Flavanoids                      0.652692        -0.172379  0.543479
0.787194
Nonflavanoid_Phenols           -0.365845         0.139057 -0.262640 -
0.503270
Proanthocyanins                 1.000000        -0.025250  0.295544
0.519067
Color_Intensity                -0.025250         1.000000 -0.521813 -
0.428815
Hue                             0.295544        -0.521813  1.000000
0.565468
OD280                           0.519067        -0.428815  0.565468
1.000000
Proline                         0.330417         0.316100  0.236183
0.312761

                       Proline
Alcohol               0.643720
Malic_Acid           -0.192011
Ash                   0.223626
Ash_Alcanity         -0.440597
Magnesium             0.393351
Total_Phenols         0.498115
Flavanoids            0.494193
Nonflavanoid_Phenols -0.311385
Proanthocyanins       0.330417
Color_Intensity       0.316100
Hue                   0.236183
OD280                 0.312761
Proline               1.000000
```

```python
##visualization of individual features
import matplotlib.pyplot as plt
import seaborn as sns
fig, axes = plt.subplots(nrows=5, ncols=3, figsize=(12, 12))
variables = df.columns
axes = axes.flatten()
# Loop through each variable and create subplots
for i, variable in enumerate(variables):
    ax = axes[i]
    sns.histplot(df[variable], ax=ax, kde=True, bins=20)
```

```
    ax.set_title(f'Distribution of {variable}')
    ax.set_xlabel(variable)
    ax.set_ylabel('Frequency')

for j in range(i + 1, len(axes)):
    fig.delaxes(axes[j])

plt.subplots_adjust(wspace=0.4, hspace=0.8)
```




```
##scaling the dats
from sklearn.preprocessing import StandardScaler
```

```
scaler = StandardScaler()
scaled_data = scaler.fit_transform(df)

scaled_data
```

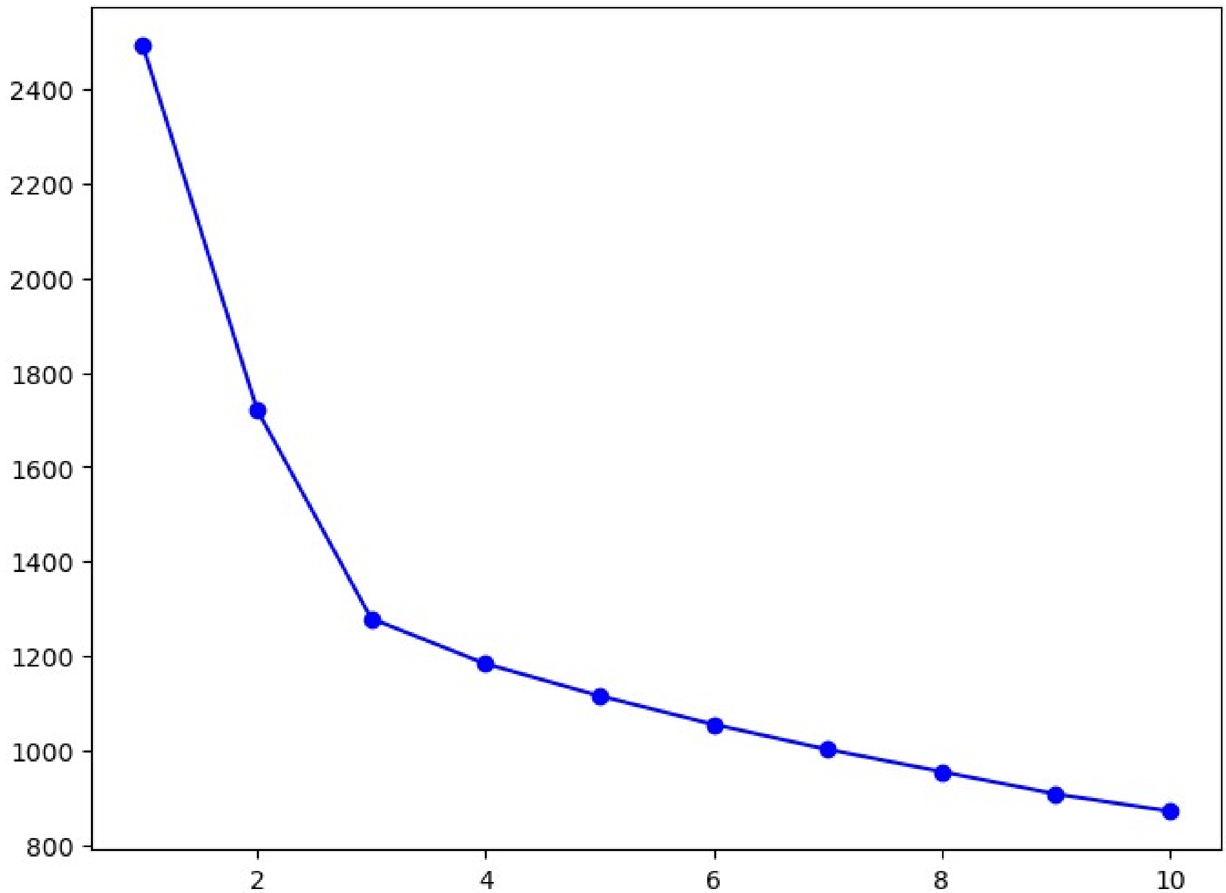
```
array([[ 1.51861254, -0.5622498 ,  0.23205254, ...,  1.84791957,
         1.01300893,  1.1928401 ],
       [ 0.24628963, -0.49941338, -0.82799632, ...,  1.1134493 ,
         0.96524152,  1.1928401 ],
       [ 0.19687903,  0.02123125,  1.10933436, ...,  0.78858745,
         1.39514818,  1.1928401 ],
       ...,
       [ 0.33275817,  1.74474449, -0.38935541, ..., -1.48544548,
         0.28057537, -1.33484488],
       [ 0.20923168,  0.22769377,  0.01273209, ..., -1.40069891,
         0.29649784, -1.33484488],
       [ 1.39508604,  1.58316512,  1.36520822, ..., -1.42894777,
        -0.59516041, -1.33484488]])
```

```
##performing k-means clustering
from sklearn.cluster import KMeans
x=scaled_data

##elbow method to calculate the optimal number of clusters
sse = []  # Initialize the list to store SSE values
k_range = range(1, 11)  # Define the range of K values

for k in k_range:
    kmeans = KMeans(n_clusters=k, init='k-means++', random_state=1)
    kmeans.fit(x)
    sse.append(kmeans.inertia_)

plt.figure(figsize=(8, 6))
plt.plot(k_range, sse, marker='o', linestyle='-', color='b')
```

```
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
```

```
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
```

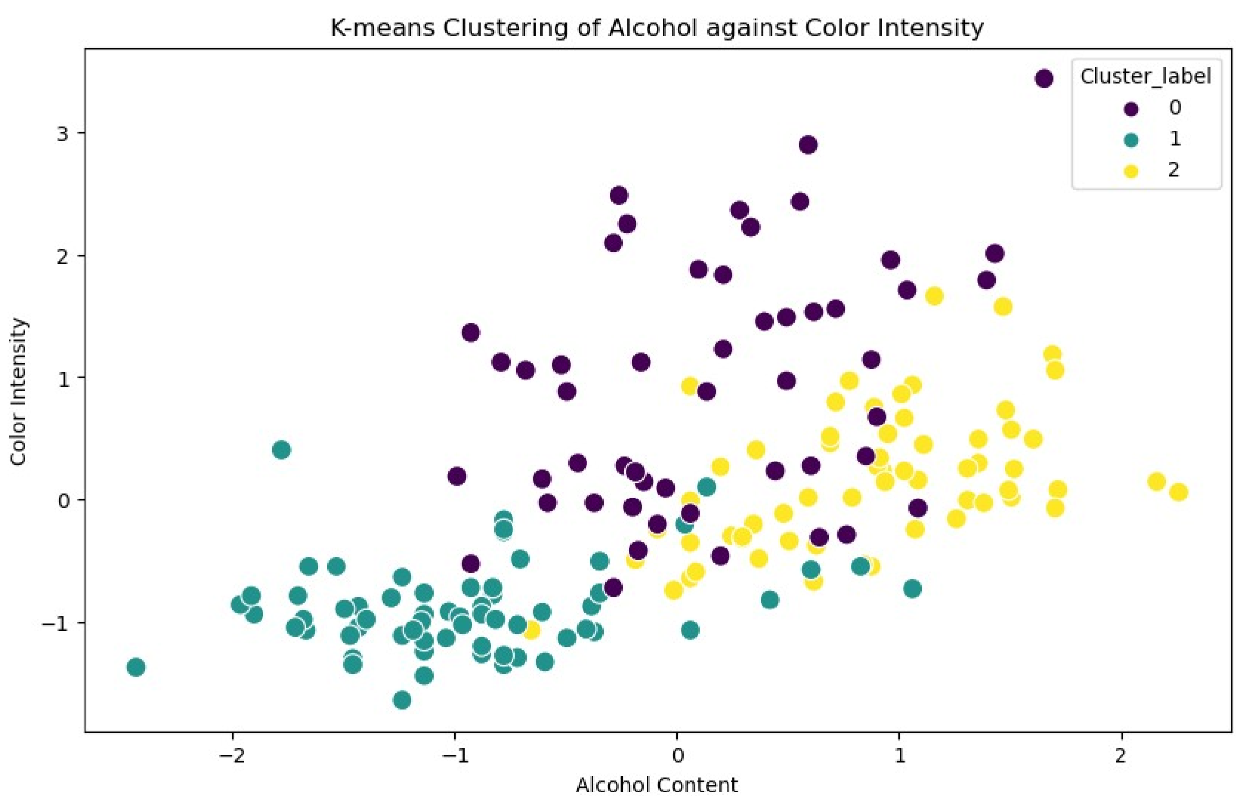
```
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
  warnings.warn(

[<matplotlib.lines.Line2D at 0x23ae54a5810>]
```



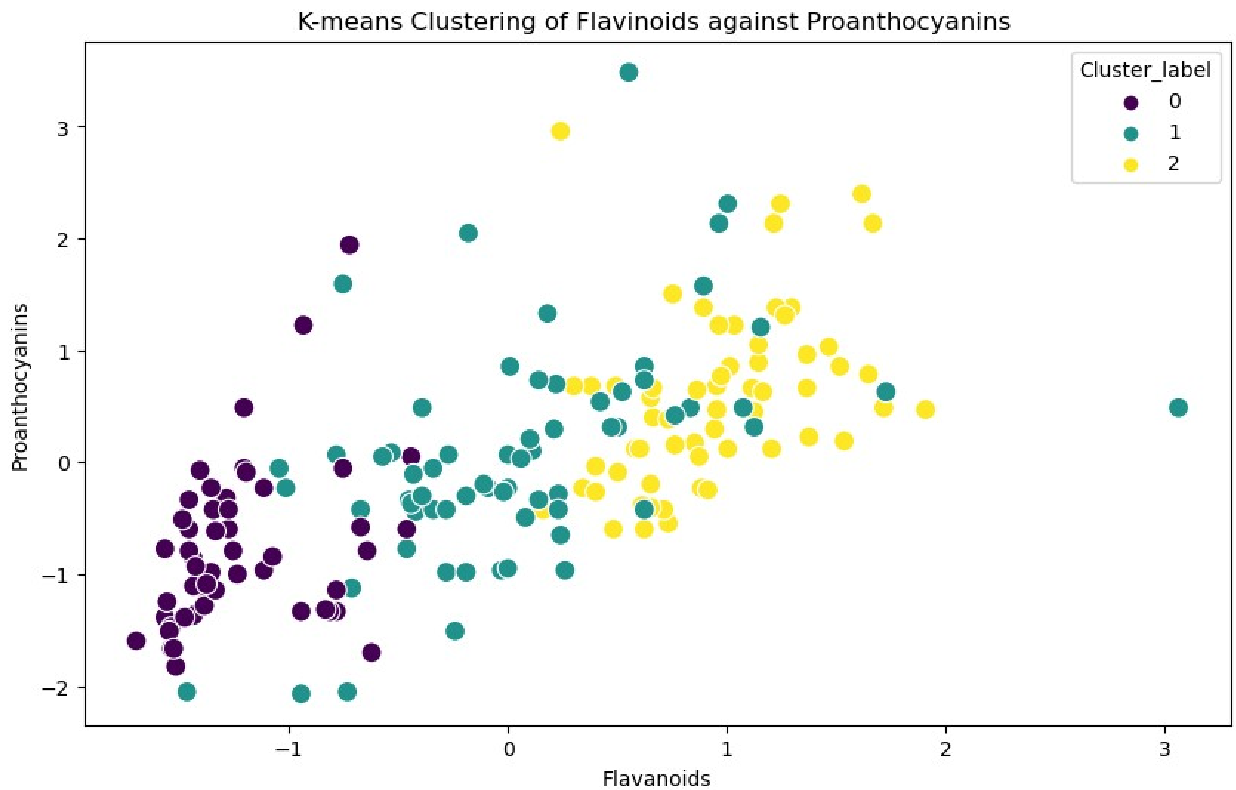

```
##converting scaled data into a dataframe and appendinf the cluster
label
df_scaled = pd.DataFrame(data=scaled_data, columns=df.columns)

labels=df_scaled['cluster_label'] = kmeans.fit_predict(x)
labels

C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\USER\anaconda3\lib\site-packages\sklearn\cluster\
_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
  warnings.warn(

array([2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
```

```
2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 0, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 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])
```

```
##visualizing the cluster for alcohol against color intensity
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df_scaled, x='Alcohol', y='Color_Intensity',
hue='cluster_label', palette='viridis', s=100)
plt.title('K-means Clustering of Alcohol against Color Intensity')
plt.xlabel('Alcohol Content')
plt.ylabel('Color Intensity')
plt.legend(title='Cluster_label')
plt.show()
```



```
##visualizing the cluster for alcohol against color intensity
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df_scaled, x='Flavanoids', y='Proanthocyanins',
hue='cluster_label', palette='viridis', s=100)
plt.title('K-means Clustering of Flavinoids against Proanthocyanins ')
plt.xlabel('Flavanoids')
plt.ylabel('Proanthocyanins')
plt.legend(title='Cluster_label')
plt.show()
```




```
#dropping the cluster_label
data=df_scaled.drop('cluster_label',axis=1)
data
```

```
       Alcohol  Malic_Acid       Ash  Ash_Alcanity  Magnesium  
Total_Phenols  \
0     1.518613   -0.562250  0.232053     -1.169593   1.913905   
0.808997   
1     0.246290   -0.499413 -0.827996     -2.490847   0.018145   
0.568648   
2     0.196879    0.021231  1.109334     -0.268738   0.088358   
0.808997   
3     1.691550   -0.346811  0.487926     -0.809251   0.930918   
2.491446   
4     0.295700    0.227694  1.840403      0.451946   1.281985
```

```
0.808997
..        ...         ...       ...           ...        ...
...
173  0.876275    2.974543  0.305159      0.301803  -0.332922      -
0.985614
174  0.493343    1.412609  0.414820      1.052516   0.158572      -
0.793334
175  0.332758    1.744744 -0.389355      0.151661   1.422412      -
1.129824
176  0.209232    0.227694  0.012732      0.151661   1.422412      -
1.033684
177  1.395086    1.583165  1.365208      1.502943  -0.262708      -
0.392751

     Flavanoids  Nonflavanoid_Phenols  Proanthocyanins
Color_Intensity  \
0      1.034819             -0.659563         1.224884
0.251717
1      0.733629             -0.820719        -0.544721        -
0.293321
2      1.215533             -0.498407         2.135968
0.269020
3      1.466525             -0.981875         1.032155
1.186068
4      0.663351              0.226796         0.401404        -
0.319276
..          ...                   ...              ...              ..
.
173   -1.424900              1.274310        -0.930179
1.142811
174   -1.284344              0.549108        -0.316950
0.969783
175   -1.344582              0.549108        -0.422075
2.224236
176   -1.354622              1.354888        -0.229346
1.834923
177   -1.274305              1.596623        -0.422075
1.791666

          Hue     OD280   Proline
0    0.362177  1.847920  1.013009
1    0.406051  1.113449  0.965242
2    0.318304  0.788587  1.395148
3   -0.427544  1.184071  2.334574
4    0.362177  0.449601 -0.037874
..        ...       ...       ...
173 -1.392758 -1.231206 -0.021952
174 -1.129518 -1.485445  0.009893
175 -1.612125 -1.485445  0.280575
```

```
176 -1.568252 -1.400699  0.296498
177 -1.524378 -1.428948 -0.595160

[178 rows x 13 columns]

##performing statistical tests to measure quality of the clusters
##silhouette test
from sklearn.metrics import silhouette_score

silhouette_avg = silhouette_score(data, labels)
silhouette_avg

0.28594199657074876

#The silhouette score is near 0, therefore indicates the clusters are
overlapping, as shown in the visualizations.

#davies_bouldin_score
from sklearn.metrics import davies_bouldin_score
db_index = davies_bouldin_score(data, labels)
db_index

1.391793832317738

##The DBI is rather high and also indicates that the clusters are not
distinct and overlap.
```