Import Libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

Load Dataset

```
from sklearn import datasets

# Load dataset from scikit-learn
wine = datasets.load_wine()
X = wine.data    # Feature data (independent) for machine learning
models
y = wine.target  # The target (dependent) label to be predicted

# Convert feature and target data into DataFrame
df_X = pd.DataFrame(X, columns=wine.feature_names)
df_y = pd.Series(y, name='target')

df_X
```

```
     alcohol  malic_acid   ash  alcalinity_of_ash  magnesium
total_phenols  \
0      14.23        1.71  2.43               15.6      127.0
2.80
1      13.20        1.78  2.14               11.2      100.0
2.65
2      13.16        2.36  2.67               18.6      101.0
2.80
3      14.37        1.95  2.50               16.8      113.0
3.85
4      13.24        2.59  2.87               21.0      118.0
2.80
..       ...         ...   ...                ...        ...
...
173    13.71        5.65  2.45               20.5       95.0
1.68
174    13.40        3.91  2.48               23.0      102.0
1.80
175    13.27        4.28  2.26               20.0      120.0
1.59
176    13.17        2.59  2.37               20.0      120.0
1.65
177    14.13        4.10  2.74               24.5       96.0
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/od315_of_diluted_wines  proline
0                            3.92   1065.0
1                            3.40   1050.0
2                            3.17   1185.0
3                            3.45   1480.0
4                            2.93    735.0
..                            ...      ...
173                          1.74    740.0
174                          1.56    750.0
175                          1.56    835.0
176                          1.62    840.0
177                          1.60    560.0

[178 rows x 13 columns]

df_y

0      0
1      0
2      0
3      0
4      0
      ..
173    2
```

```
174    2
175    2
176    2
177    2
Name: target, Length: 178, dtype: int32
```

```
# Combine features and targets in one DataFrame
df = pd.concat([df_X, df_y], axis=1)
df.head(10)
```

```
   alcohol  malic_acid   ash  alcalinity_of_ash  magnesium
total_phenols  \
0    14.23        1.71  2.43               15.6      127.0
2.80
1    13.20        1.78  2.14               11.2      100.0
2.65
2    13.16        2.36  2.67               18.6      101.0
2.80
3    14.37        1.95  2.50               16.8      113.0
3.85
4    13.24        2.59  2.87               21.0      118.0
2.80
5    14.20        1.76  2.45               15.2      112.0
3.27
6    14.39        1.87  2.45               14.6       96.0
2.50
7    14.06        2.15  2.61               17.6      121.0
2.60
8    14.83        1.64  2.17               14.0       97.0
2.80
9    13.86        1.35  2.27               16.0       98.0
2.98

   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
5        3.39                  0.34             1.97             6.75
1.05
6        2.52                  0.30             1.98             5.25
1.02
7        2.51                  0.31             1.25             5.05
```

```
1.06
8        2.98                  0.29             1.98              5.20
1.08
9        3.15                  0.22             1.85              7.22
1.01

   od280/od315_of_diluted_wines  proline  target
0                          3.92   1065.0       0
1                          3.40   1050.0       0
2                          3.17   1185.0       0
3                          3.45   1480.0       0
4                          2.93    735.0       0
5                          2.85   1450.0       0
6                          3.58   1290.0       0
7                          3.58   1295.0       0
8                          2.85   1045.0       0
9                          3.55   1045.0       0
```

Exploratory Data Analysis

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 178 entries, 0 to 177
Data columns (total 14 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   alcalinity_of_ash             178 non-null    float64
 4   magnesium                     178 non-null    float64
 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/od315_of_diluted_wines  178 non-null    float64
 12  proline                       178 non-null    float64
 13  target                        178 non-null    int32  
dtypes: float64(13), int32(1)
memory usage: 18.9 KB

df.isna().sum()

alcohol                         0
malic_acid                      0
ash                             0
alcalinity_of_ash               0
```

```
magnesium                       0
total_phenols                   0
flavanoids                      0
nonflavanoid_phenols            0
proanthocyanins                 0
color_intensity                 0
hue                             0
od280/od315_of_diluted_wines    0
proline                         0
target                          0
dtype: int64
```

```
df['target'].unique()
```

```
array([0, 1, 2])
```

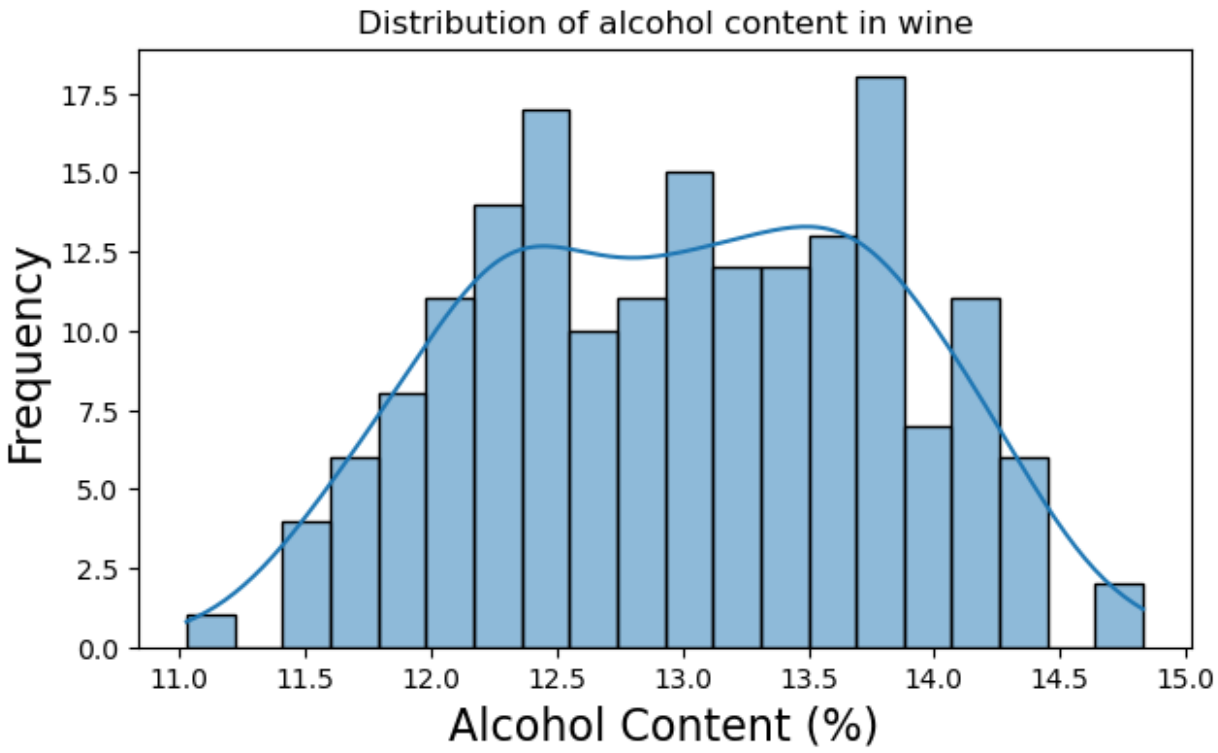
```
df.describe()
```

```
          alcohol  malic_acid         ash  alcalinity_of_ash
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/od315_of_diluted_wines
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

           target
count  178.000000
mean     0.938202
std      0.775035
min      0.000000
25%      0.000000
50%      1.000000
75%      2.000000
max      2.000000
```

Data Visualization

```
# Plot Distribution of alcohol content
plt.figure(figsize=(7, 4))
sns.histplot(df_X['alcohol'], kde=True, bins=20)
plt.title("Distribution of alcohol content in wine")
plt.xlabel("Alcohol Content (%)", fontsize=16)
plt.ylabel("Frequency", fontsize=16)
plt.show()
```



Train Test Split

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(df_X, df_y,
test_size=0.2, random_state=42)
```

Logistic Regression

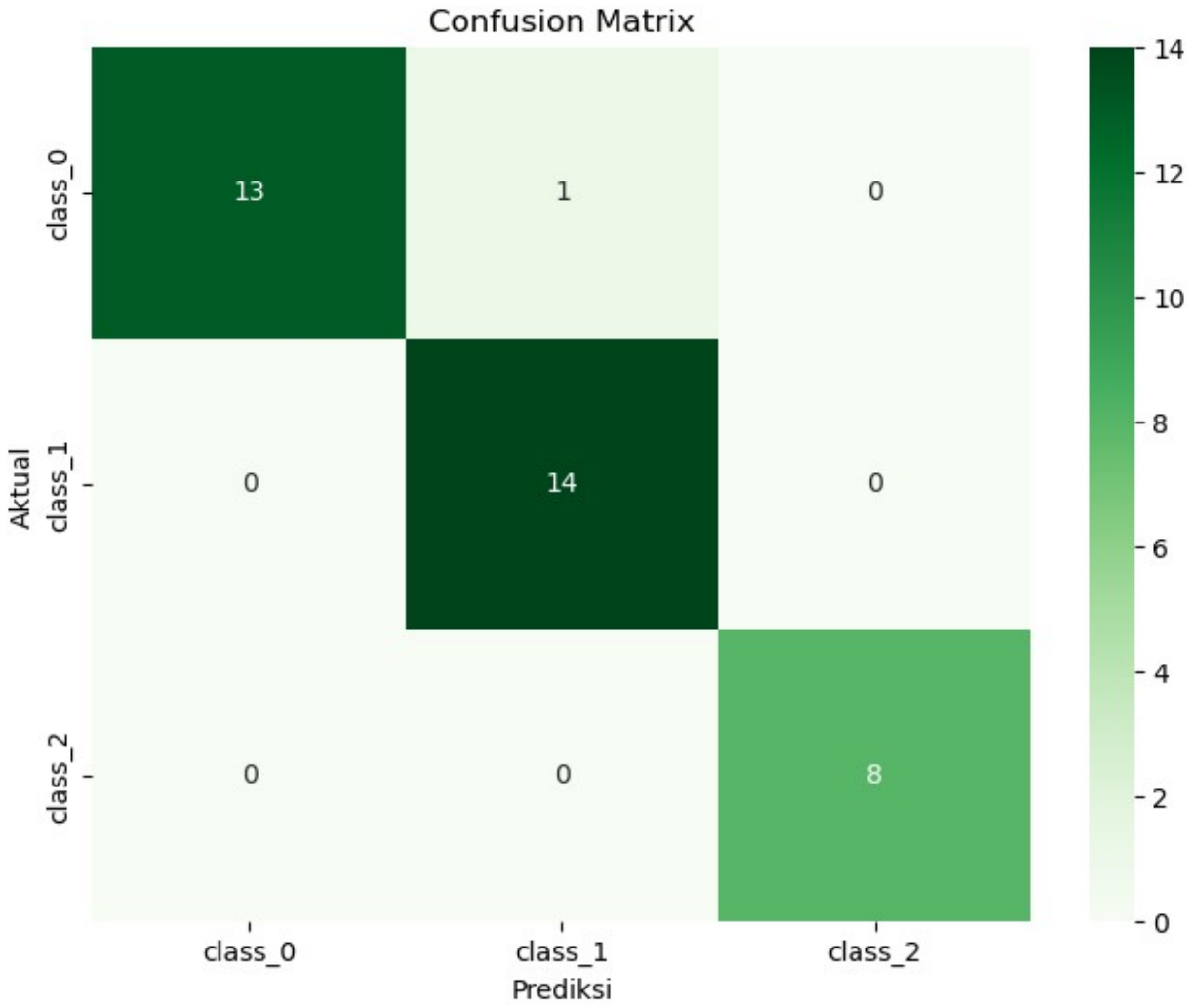
```
from sklearn.linear_model import LogisticRegression

model_lr = LogisticRegression(random_state=42)
model_lr.fit(X_train, y_train)
```

```
LogisticRegression(random_state=42)
```

```
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

y_pred_lr = model_lr.predict(X_test)
accuracy = accuracy_score(y_test, y_pred_lr)

print(f"Akurasi Logistic Regression: {accuracy * 100:.2f}%")
```

```
Akurasi Logistic Regression: 97.22%
```

```
# Confusion Matrix
cm_lr = confusion_matrix(y_test, y_pred_lr)

# Plot Confusion Matrix using seaborn
plt.figure(figsize=(8, 6))
sns.heatmap(cm_lr, annot=True, fmt="d", cmap="Greens",
            xticklabels=wine.target_names,
yticklabels=wine.target_names)
plt.xlabel("Prediksi")
plt.ylabel("Aktual")
plt.title("Confusion Matrix")
plt.show()
```




```
from sklearn.metrics import accuracy_score, classification_report

print(classification_report(y_test, y_pred_lr,
target_names=wine.target_names))
```

```
              precision    recall  f1-score   support

     class_0       1.00      0.93      0.96        14
     class_1       0.93      1.00      0.97        14
     class_2       1.00      1.00      1.00         8

    accuracy                           0.97        36
   macro avg       0.98      0.98      0.98        36
weighted avg       0.97      0.97      0.97        36
```

Gradient Boosting

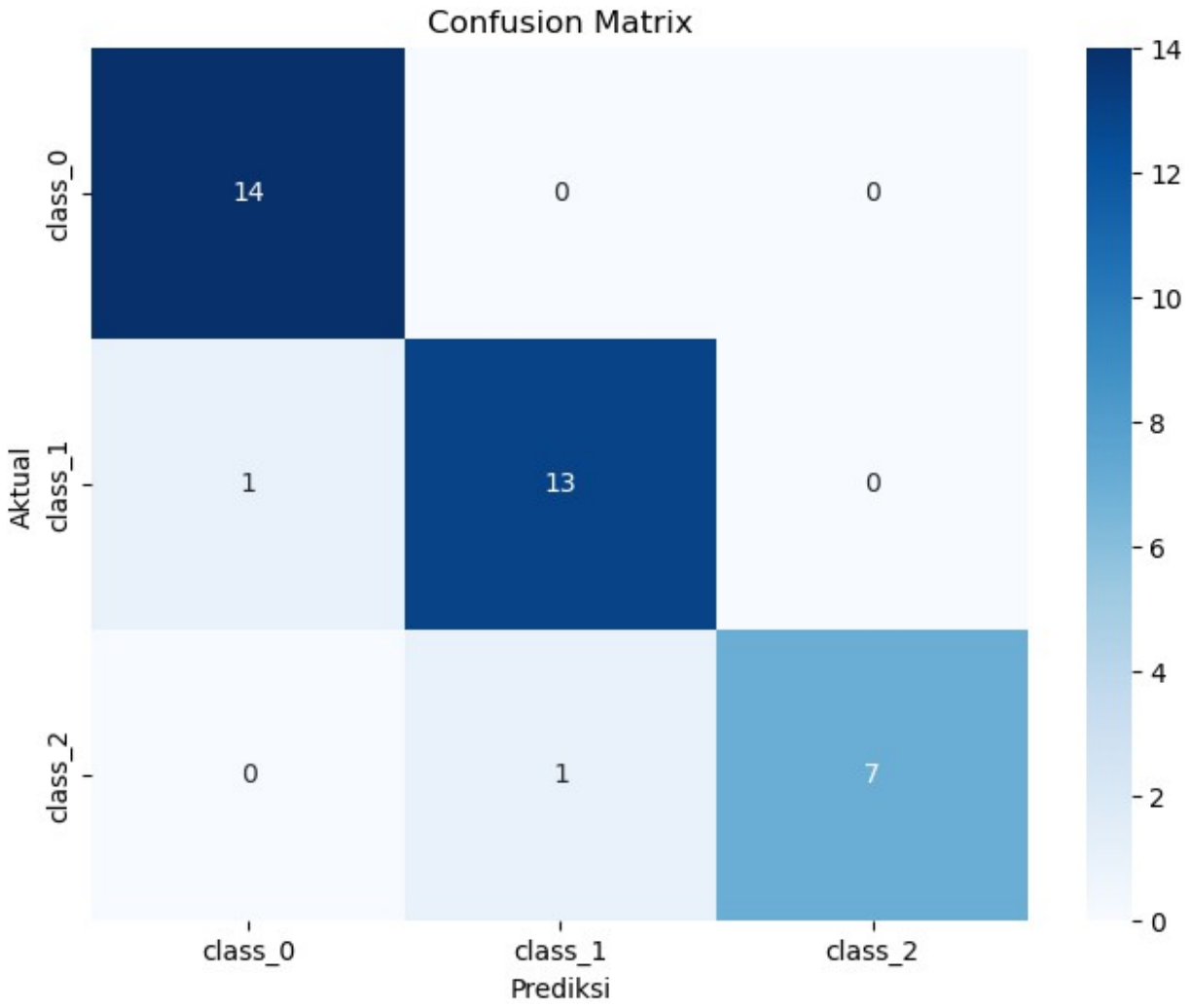
```
from sklearn.ensemble import GradientBoostingClassifier

model_gb = GradientBoostingClassifier(random_state=42)
model_gb.fit(X_train, y_train)
```

```
GradientBoostingClassifier(random_state=42)
```

```
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

y_pred_gb = model_gb.predict(X_test)
accuracy = accuracy_score(y_test, y_pred_gb)

print(f"Akurasi Gradient Boosting: {accuracy_score(y_test, y_pred_gb)
* 100:.2f}%")
```

```
Akurasi Gradient Boosting: 94.44%
```

```
# Confusion Matrix
cm_gb = confusion_matrix(y_test, y_pred_gb)

# Plot Confusion Matrix using seaborn
plt.figure(figsize=(8, 6))
sns.heatmap(cm_gb, annot=True, fmt="d", cmap="Blues",
            xticklabels=wine.target_names,
yticklabels=wine.target_names)
plt.xlabel("Prediksi")
plt.ylabel("Aktual")
plt.title("Confusion Matrix")
plt.show()
```




```
from sklearn.metrics import accuracy_score, classification_report

print(classification_report(y_test, y_pred_gb,
target_names=wine.target_names))
```

```
              precision    recall  f1-score   support

     class_0       0.93      1.00      0.97        14
     class_1       0.93      0.93      0.93        14
     class_2       1.00      0.88      0.93         8

    accuracy                           0.94        36
   macro avg       0.95      0.93      0.94        36
weighted avg       0.95      0.94      0.94        36
```

Support vector machine (SVM)

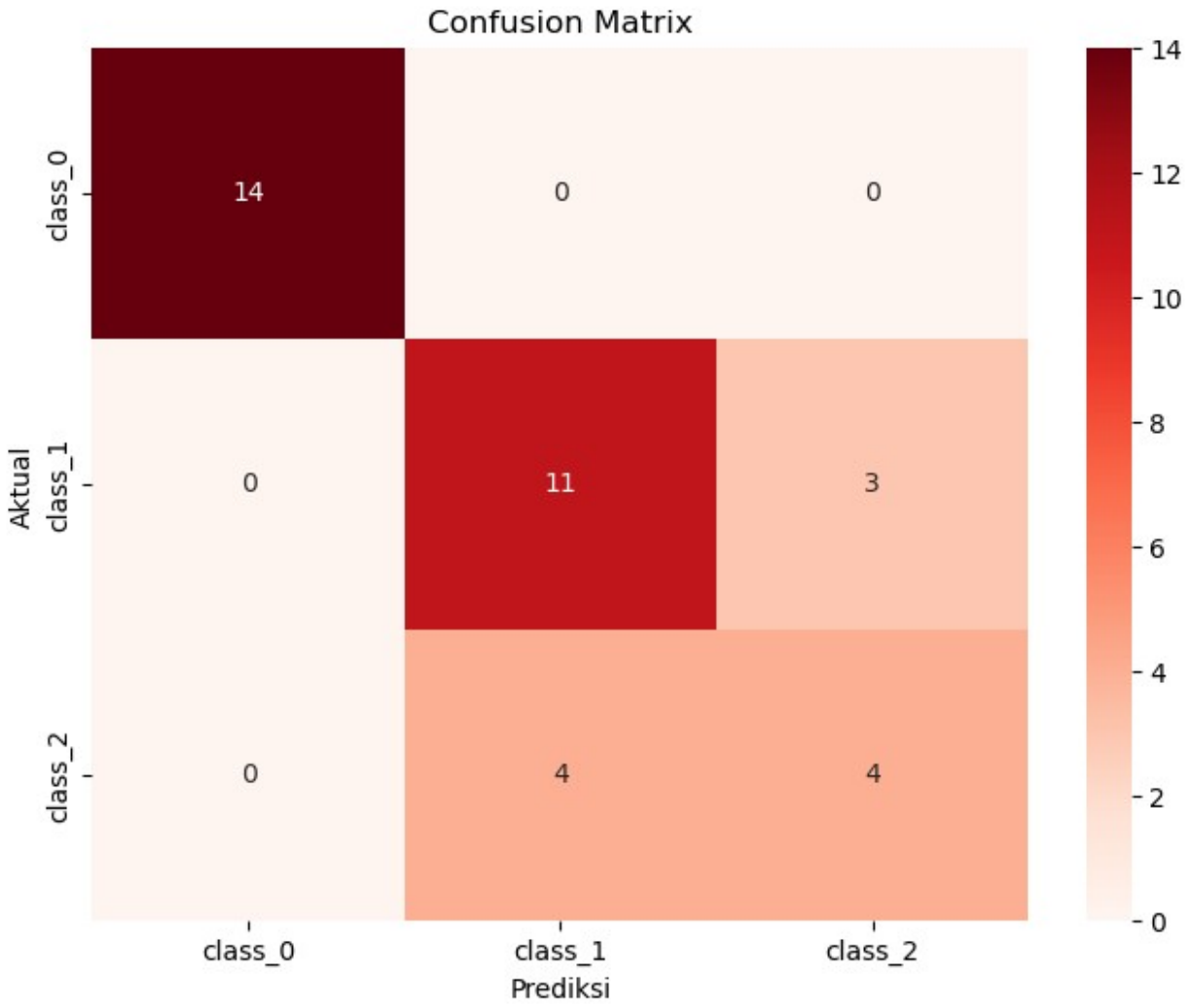
```
from sklearn.svm import SVC

model_svm = SVC(random_state=42)
model_svm.fit(X_train, y_train)
```

```
SVC(random_state=42)
```

```
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

y_pred_svm = model_svm.predict(X_test)
accuracy = accuracy_score(y_test, y_pred_svm)

print(f"Akurasi Support Vector Machine (SVM): {accuracy_score(y_test,
y_pred_svm) * 100:.2f}%")
```

```
Akurasi Support Vector Machine (SVM): 80.56%
```

```
# Confusion Matrix
cm_svm = confusion_matrix(y_test, y_pred_svm)

# Plot Confusion Matrix using seaborn
plt.figure(figsize=(8, 6))
sns.heatmap(cm_svm, annot=True, fmt="d", cmap="Reds",
            xticklabels=wine.target_names,
yticklabels=wine.target_names)
plt.xlabel("Prediksi")
plt.ylabel("Aktual")
plt.title("Confusion Matrix")
plt.show()
```




```
from sklearn.metrics import accuracy_score, classification_report

print(classification_report(y_test, y_pred_svm,
target_names=wine.target_names))
```

```
              precision    recall  f1-score   support

     class_0       1.00      1.00      1.00        14
     class_1       0.73      0.79      0.76        14
     class_2       0.57      0.50      0.53         8

    accuracy                           0.81        36
   macro avg       0.77      0.76      0.76        36
weighted avg       0.80      0.81      0.80        36
```