Wine Classification

Machine Learning Project

Presented by Afifah Hadi Lestari

Project Overview

In this project, I built a Machine Learning model to classify wine types based on their chemical characteristics using the Wine Dataset from Scikit-Learn. The dataset consists of 178 samples with 13 features, like the alcohol content and flavonoids. I compared three algorithms, it is Logistic Regression, Gradient Boosting, and Support Vector Machine (SVM), to determine the best performing model. The process started with data exploration and visualization, and then the models were trained and evaluated using accuracy, confusion matrix, and classification report.

Table of Contents

01

Data Preparation

Load the Wine dataset from Scikit-Learn, explored and checked to ensure data quality. Statistical analysis is performed to understand the distribution of features, and the dataset is divided into 80% training and 20% testing.

Data Preparation

The Wine dataset loaded from Scikit-learn, and then converted into DataFrame for easy processing.

```
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')
```

View of some DataFrame features.

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity	hue
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	0.28	2.29	5.64	1.04
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	0.26	1.28	4.38	1.05
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	0.30	2.81	5.68	1.03
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	0.24	2.18	7.80	0.86
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	0.39	1.82	4.32	1.04
5	14.20	1.76	2.45	15.2	112.0	3.27	3.39	0.34	1.97	6.75	1.05
6	14.39	1.87	2.45	14.6	96.0	2.50	2.52	0.30	1.98	5.25	1.02
7	14.06	2.15	2.61	17.6	121.0	2.60	2.51	0.31	1.25	5.05	1.06
8	14.83	1.64	2.17	14.0	97.0	2.80	2.98	0.29	1.98	5.20	1.08
9	13.86	1.35	2.27	16.0	98.0	2.98	3.15	0.22	1.85	7.22	1.01

Describes general information about the Dataset.

```
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
```

Data Preparation

From this output, no missing values in the DataFrame.

```
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
```

Describe the statistical results of the Dataset from several features.

```
df.describe()
```

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity	hue
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000
mean	13.000618	2.336348	2.366517	19.494944	99.741573	2.295112	2.029270	0.361854	1.590899	5.058090	0.957449
std	0.811827	1.117146	0.274344	3.339564	14.282484	0.625851	0.998859	0.124453	0.572359	2.318286	0.228572
min	11.030000	0.740000	1.360000	10.600000	70.000000	0.980000	0.340000	0.130000	0.410000	1.280000	0.480000
25%	12.362500	1.602500	2.210000	17.200000	88.000000	1.742500	1.205000	0.270000	1.250000	3.220000	0.782500
50%	13.050000	1.865000	2.360000	19.500000	98.000000	2.355000	2.135000	0.340000	1.555000	4.690000	0.965000
75%	13.677500	3.082500	2.557500	21.500000	107.000000	2.800000	2.875000	0.437500	1.950000	6.200000	1.120000
max	14.830000	5.800000	3.230000	30.000000	162.000000	3.880000	5.080000	0.660000	3.580000	13.000000	1.710000

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

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

This shows the target column has three different classes, that are 0, 1, and 2. This indicates the Wine dataset is a multiclass classification problem.

02

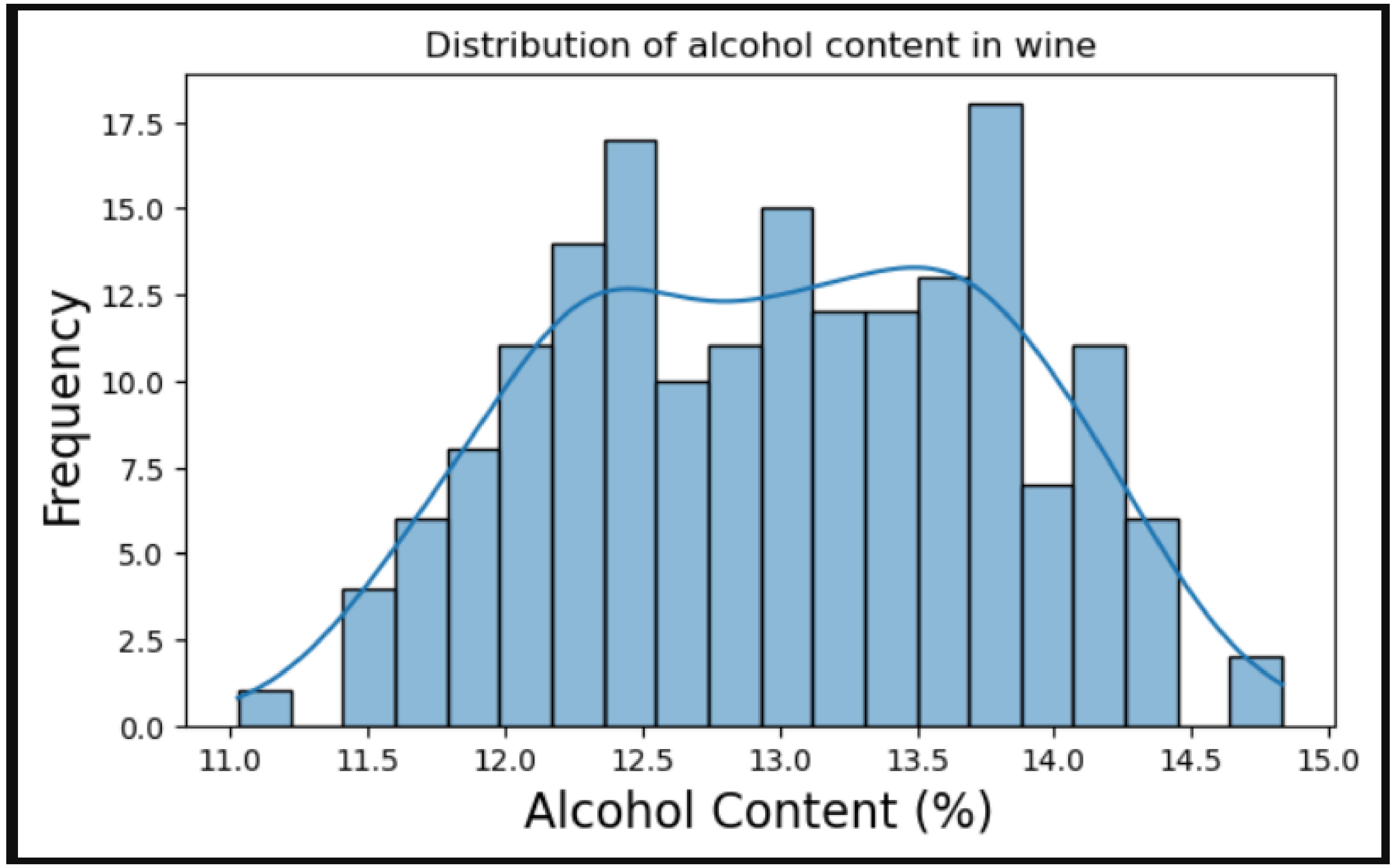
Data Visualization

Data visualization used to understand the distribution of key features and relationships between variables. For this case, one of the visualizations used is a histogram of the distribution of alcohol content in Wine, which shows how the alcohol values are spread across the Dataset.

Data Visualization

This graph shows the distribution of alcohol content in the wine dataset using a histogram with Kernel Density Estimation (KDE) to see the distribution pattern of the data. From this visualization, it can be observed that the alcohol content in the wine ranges from 11% to 14.5%, with the majority of the samples being around 12.5% - 13.5%. The distribution pattern looks close to a normal distribution, but with a slight skewness to the right. This information is useful in understanding the characteristics of wine and how alcohol content can affect its classification in Machine Learning models.

```
# 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()
```



03

Machine Learning Modeling

Train machine learning algorithms to classify wine types. The training process start with dividing the dataset into training and testing Dataset to objectively measure the model performance.

Model Training

Split the Dataset into training and testing.

```
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)
```

Gradient Boosting

```
from sklearn.ensemble import GradientBoostingClassifier

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

GradientBoostingClassifier

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

Support Vector Machine (SVM)

```
from sklearn.svm import SVC

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

SVC

```
SVC(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

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

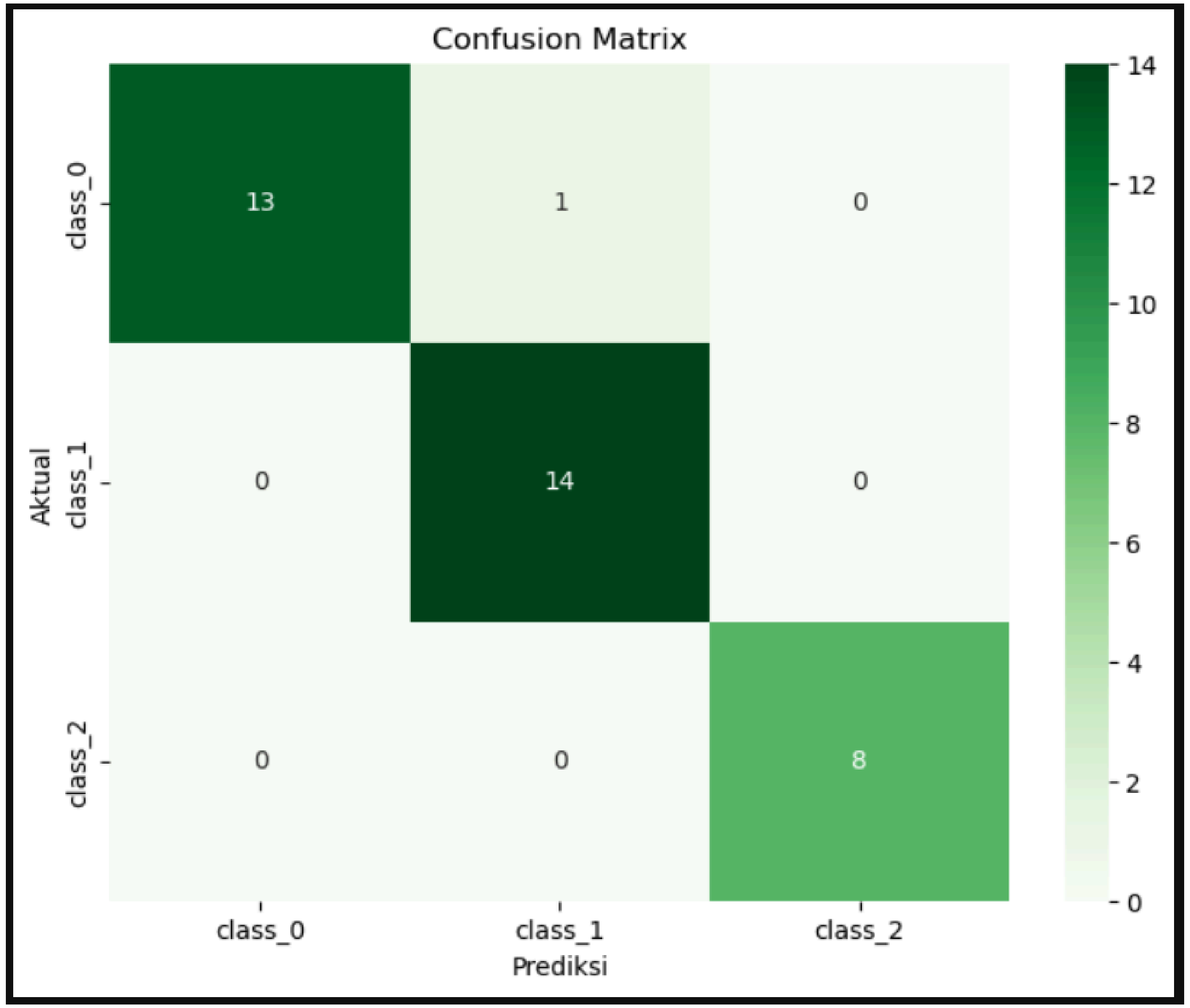
04

Model Evaluation

Model Evaluation to create the performance of machine learning models in accurately predicting data. This evaluation is important to ensure the model not only provide good performance on training data but make generalizations to new data.

Logistic Regression

Confusion Matrix



Accuracy Score

```
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}%")
```

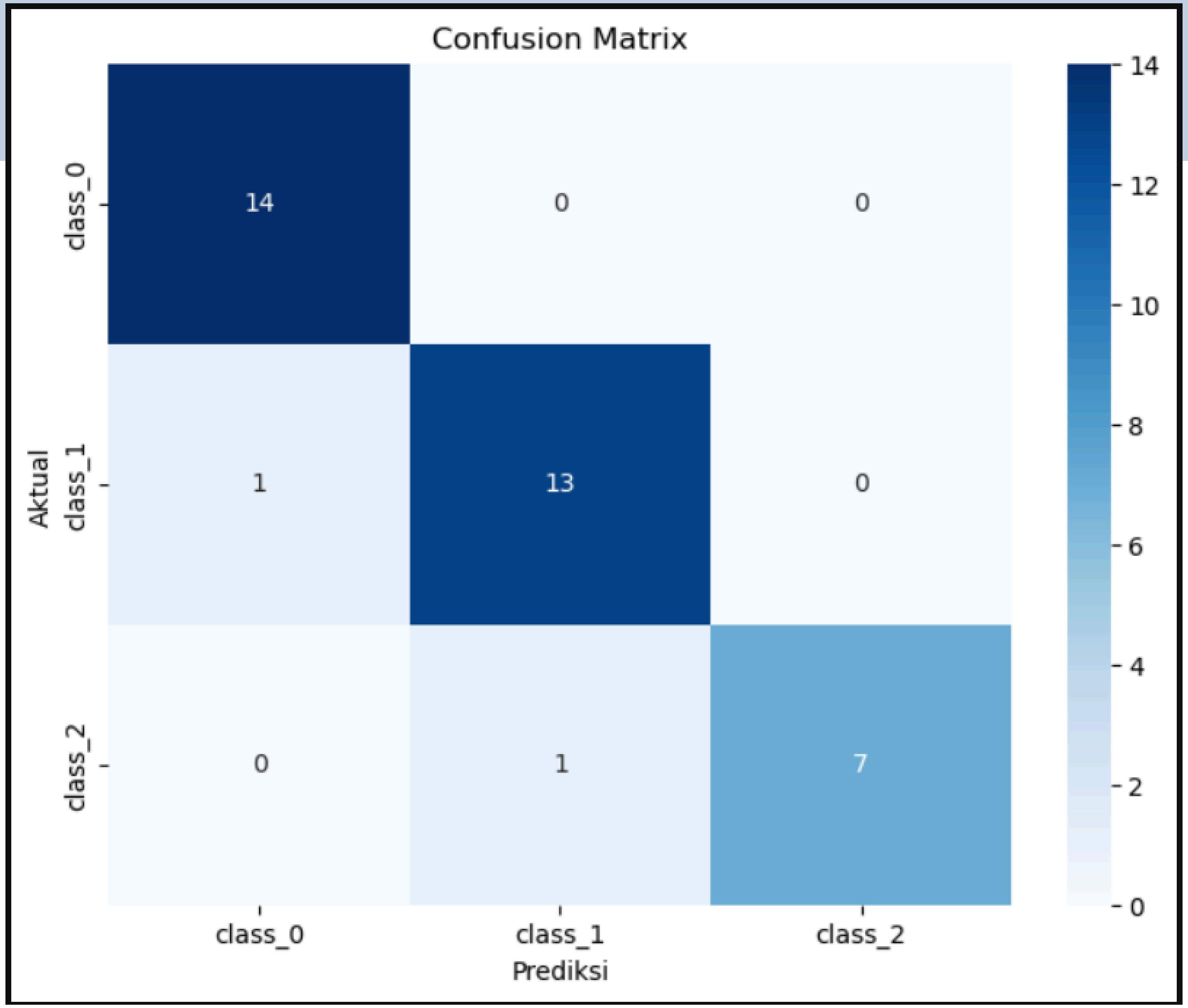
```
Akurasi Logistic Regression: 97.22%
```

Evaluation Report

	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

Confusion Matrix



Accuracy Score

```
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%
```

Evaluation Report

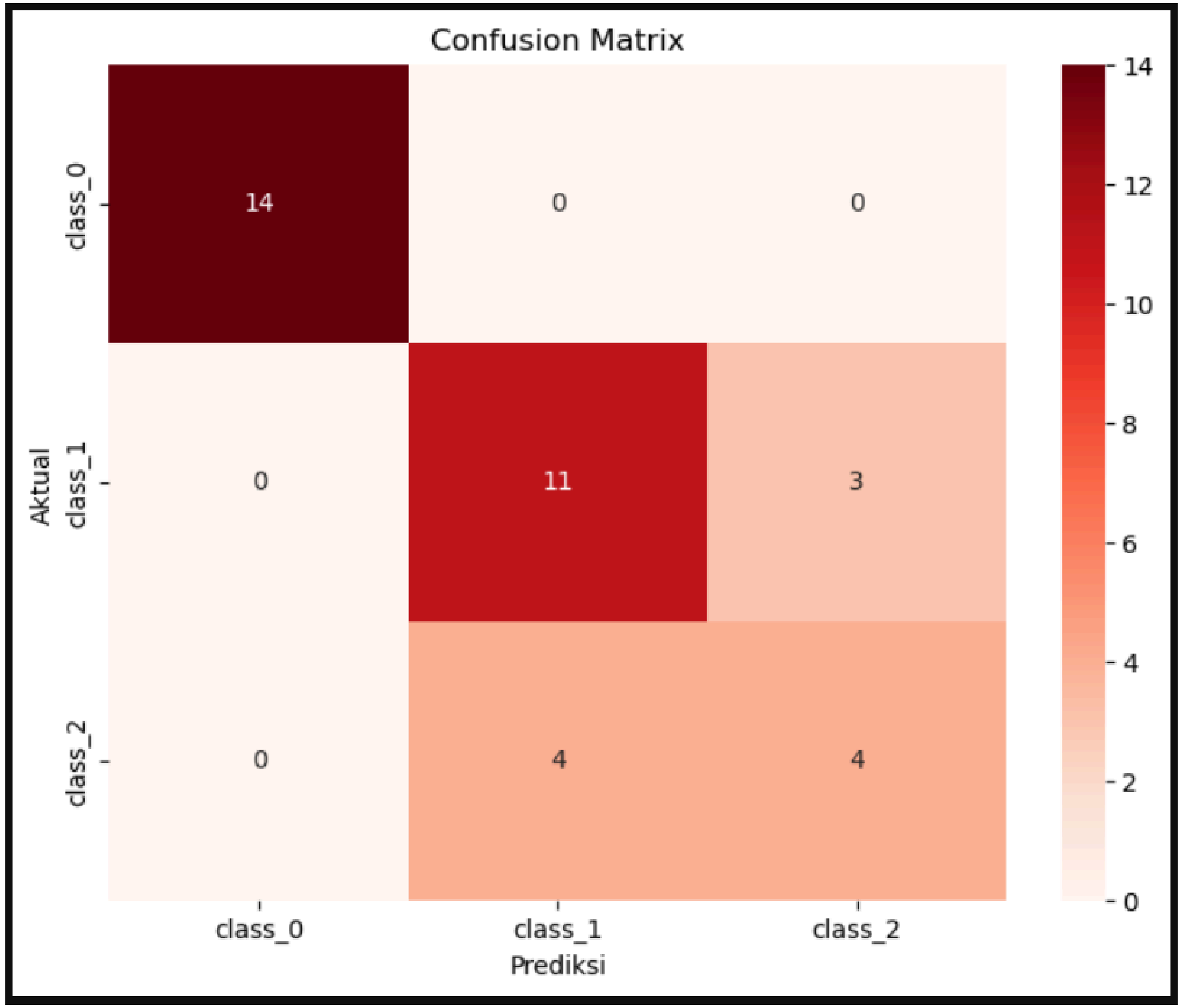
```
              precision    recall  f1-score   support

     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 machines (SVM)

Confusion Matrix



Accuracy Score

```
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%
```

Evaluation Report

	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

Conclusion

Based on the model evaluation results shown in the classification report, the classification model used demonstrates excellent performance. The model achieves an accuracy of 97%, indicating that most of its predictions are correct.

- Precision, recall, and F1-score for all three classes have high values, with some reaching 1.00, indicating that the model can accurately identify categories with minimal errors.
- Class 2 has an F1-score of 1.00, meaning the model perfectly classifies this class.
- Class 1 has a recall of 1.00, showing that all instances of this class are correctly identified without any being missed.

Overall, this model demonstrates highly optimal performance in classifying the data. However, despite these excellent results, further testing on different datasets is necessary to ensure the model's generalization ability and avoid potential overfitting.

Thank You!

Thank you for taking the time to review my work. I hope this portfolio provides valuable insights and showcases my passion for the field of machine learning.

afifahhadie11@gmail.com

https://www.linkedin.com/in/afifahhadilestari/

https://github.com/afifahhadie