: INTERNSHIP PROJECT:

B. RAVI KUMAR
VU21CSEN0500333
Ph:9391350642

1.Problem statement: -

Iris Flowers Classification: -

Project Idea: The Iris Dataset downloaded from UKI M. Repository-Download Iris Flowers Dataset. The goal of this data science project for beginners is to classify the flowers into three species-Virginia, setose, or versicolor based on the length and width of the petals and sepals.

Industry: Medicine

CODE: -

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, roc_curve, roc_auc_score, ConfusionMatrixDisplay

from sklearn.multiclass import OneVsRestClassifier

from sklearn.preprocessing import label_binarize

import matplotlib.pyplot as plt

import seaborn as sns

import numpy as np

from mpl_toolkits.mplot3d import Axes3D
```

```
# Load the dataset

df = pd.read_csv("/content/Iris.csv")


# Preprocessing

if 'Id' in df.columns:

    df = df.drop('Id', axis=1)

X = df.drop('Species', axis=1)

y = df['Species']


# Split data into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# Standardize features

scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)


# Train the logistic regression model

model = LogisticRegression(random_state=42)

model.fit(X_train, y_train)


# Make predictions on the test set

y_pred = model.predict(X_test)
```

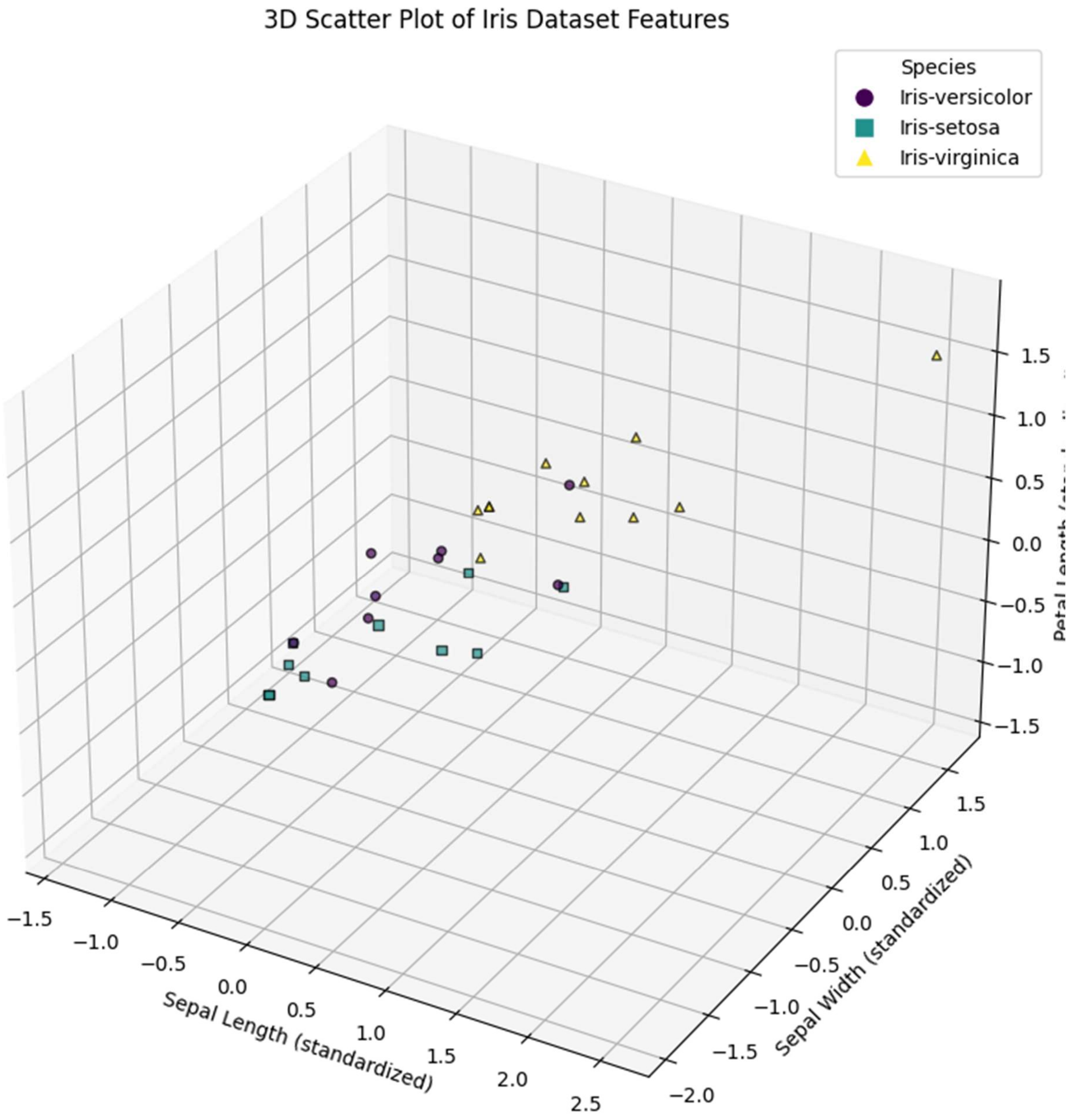
```
# Get unique species for color mapping
unique_species = y_test.unique()

# Create a color dictionary for each species
color_map = dict(zip(unique_species, plt.cm.viridis(np.linspace(0, 1, len(unique_species)))))
marker_map = dict(zip(unique_species, ['o', 's', '^']))  # Different markers for each species

# Visualization - 3D Scatter Plot
fig = plt.figure(figsize=(12, 10))
ax = fig.add_subplot(111, projection='3d')

# Extract features for 3D plotting
feature1 = X_test[:, 0]  # First feature
feature2 = X_test[:, 1]  # Second feature
feature3 = X_test[:, 2]  # Third feature

# Use color and marker mapping for scatter plot
for species in unique_species:
    indices = y_pred == species
    ax.scatter(feature1[indices], feature2[indices], feature3[indices],
               c=[color_map[species]] * sum(indices), marker=marker_map[species],
               label=species, alpha=0.7, edgecolors='k')

# Labels and title
```

```
ax.set_xlabel('Sepal Length (standardized)')
ax.set_ylabel('Sepal Width (standardized)')
ax.set_zlabel('Petal Length (standardized)')
ax.set_title('3D Scatter Plot of Iris Dataset Features')

# Legend with color and marker mapping
handles = [plt.Line2D([0], [0], marker=marker_map[species], color='w',
markerfacecolor=color_map[species],
               markersize=10, label=species) for species in unique_species]
ax.legend(handles=handles, title='Species')

plt.show()

# Pair Plot
sns.pairplot(df, hue='Species', markers=["o", "s", "D"])
plt.suptitle('Pair Plot of Iris Dataset Features', y=1.02)
plt.show()

# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
cmd = ConfusionMatrixDisplay(cm, display_labels=model.classes_)
cmd.plot(cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.show()
```

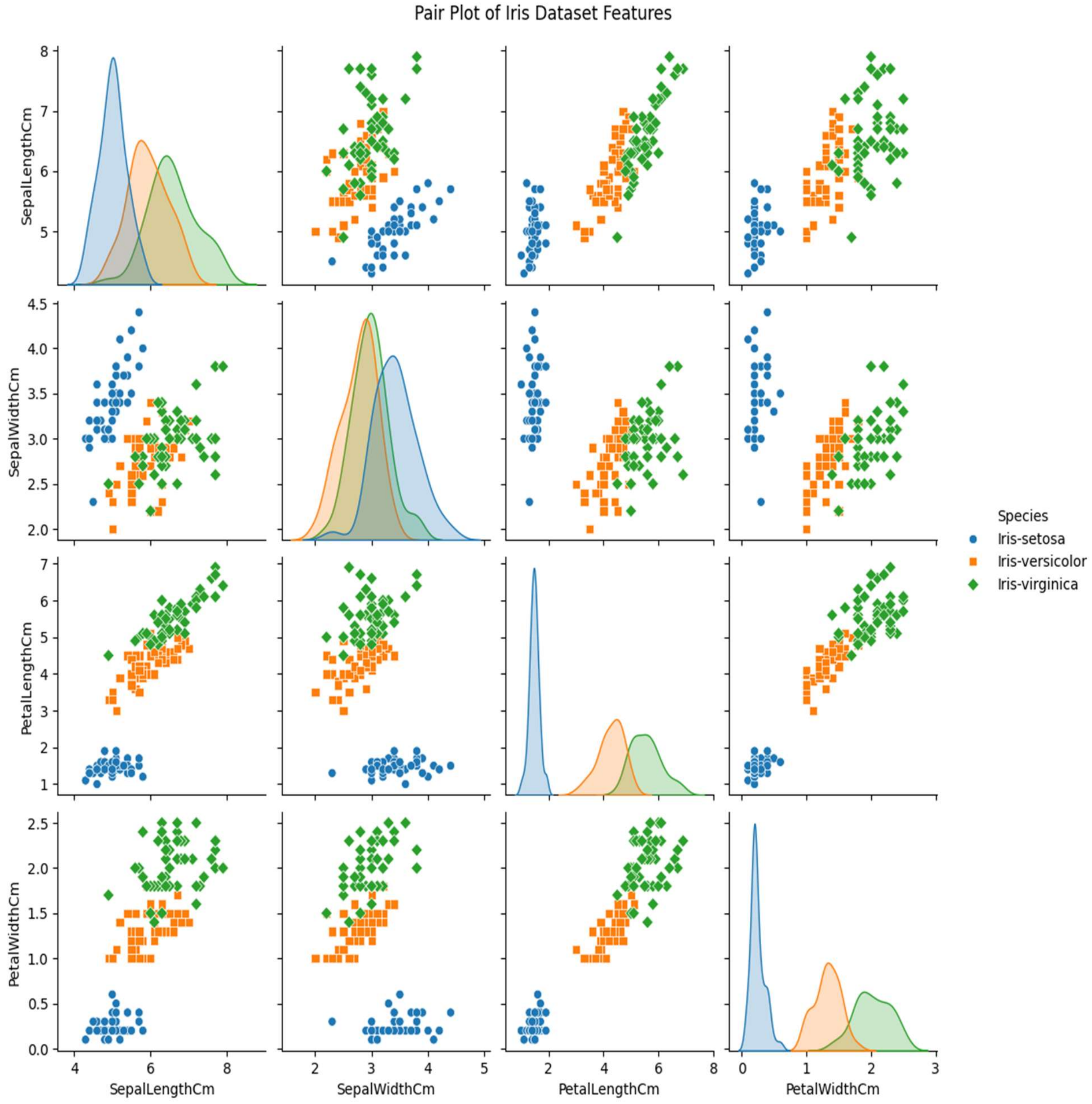
```
# ROC Curve

# Binarize the output

y_test_bin = label_binarize(y_test, classes=model.classes_)

n_classes = y_test_bin.shape[1]


# Train the OneVsRestClassifier

classifier = OneVsRestClassifier(LogisticRegression(random_state=42))

classifier.fit(X_train, y_train)

y_score = classifier.predict_proba(X_test)


# Compute ROC curve and ROC area for each class

fpr = dict()

tpr = dict()

roc_auc = dict()

for i in range(n_classes):

    fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_score[:, i])

    roc_auc[i] = roc_auc_score(y_test_bin[:, i], y_score[:, i])


# Plot ROC curve

plt.figure()

colors = ['aqua', 'darkorange', 'cornflowerblue']

for i, color in zip(range(n_classes), colors):
    plt.plot(fpr[i], tpr[i], color=color, lw=2, label=f'ROC curve of class {model.classes_[i]} (area = {roc_auc[i]:0.2f})')

plt.plot([0, 1], [0, 1], 'k--', lw=2)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve for Multiclass Classification')
plt.legend(loc="lower right")
plt.show()
# Feature Importance Plot
# Get feature importance
feature_importance = np.abs(model.coef_[0])
# Plot feature importance
plt.bar(df.columns[:-1], feature_importance)
plt.xlabel('Features')
plt.ylabel('Importance')
plt.title('Feature Importance for Logistic Regression')
plt.show()
```

OUTPUT: -

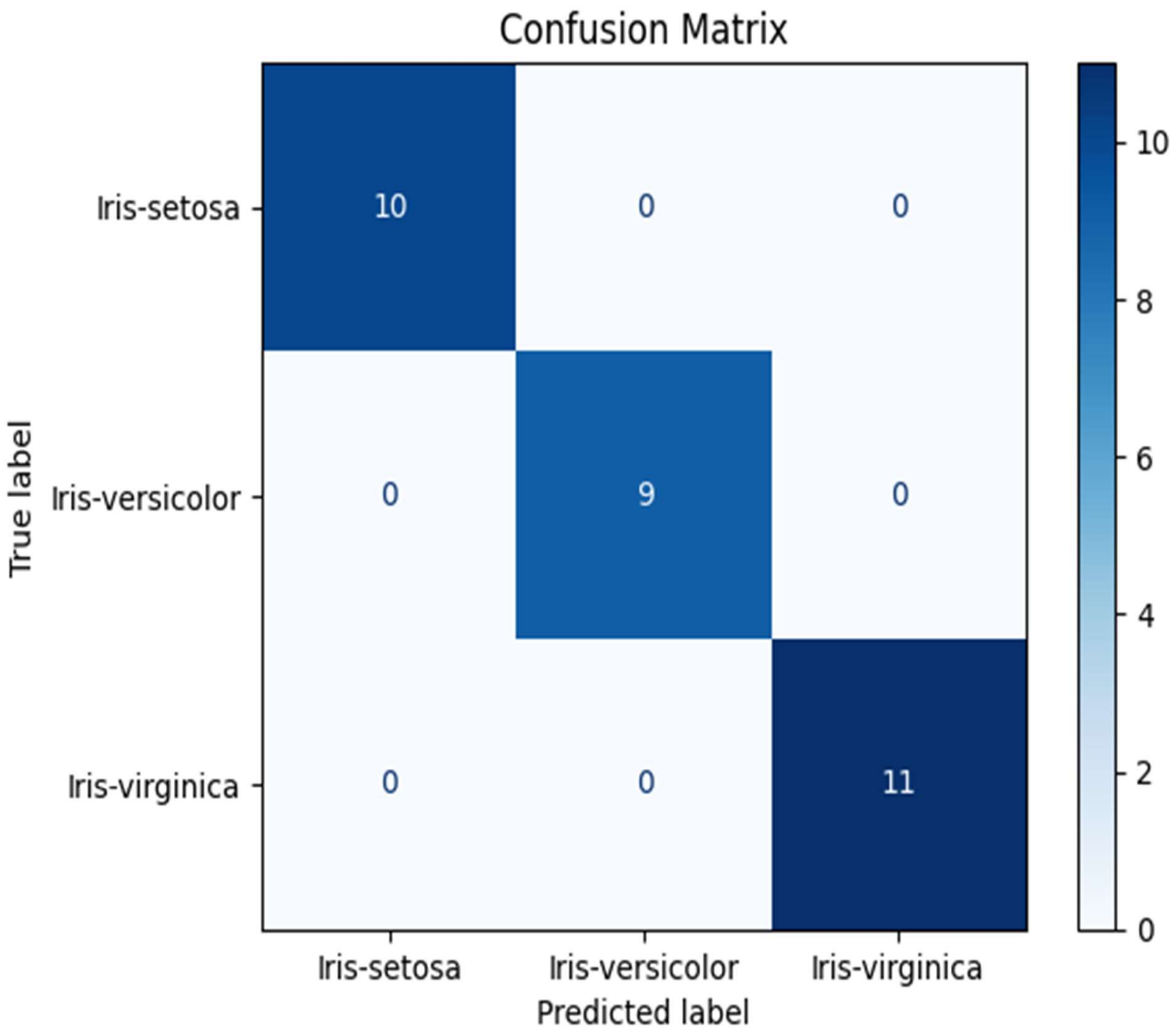
1. 3D Scatter Plot.



2.Pair Plot.

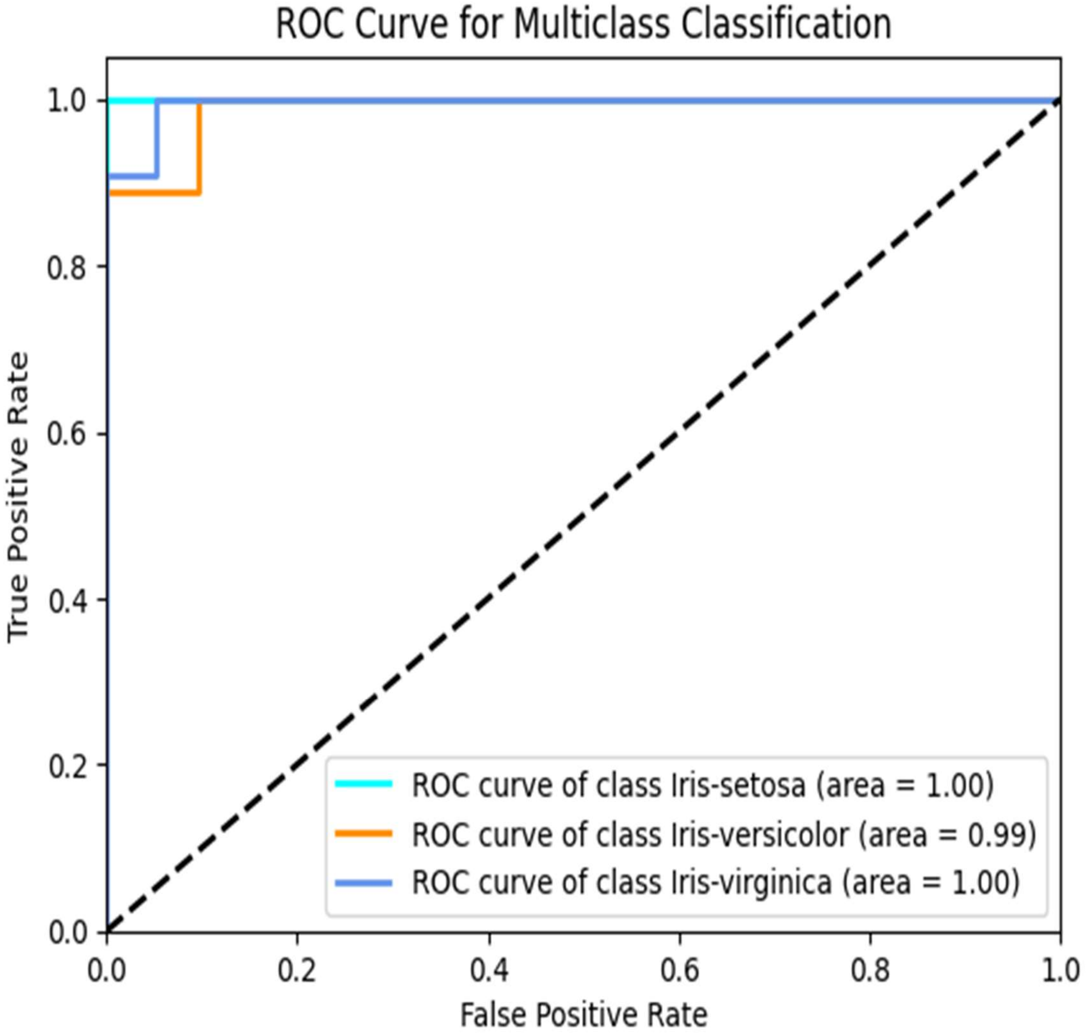

Pair Plot of Iris Dataset Features
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
8
7
6
5
4.5
4.0
3.5
3.0
2.5
2.0
7
6
5
4
3
2
1
2.5
2.0
1.5
1.0
0.5
0.0
4
6
8
2
3
4
5
2
4
6
8
0
1
2
3
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species
● Iris-setosa
■ Iris-versicolor
◆ Iris-virginica

3.Confusion Matrix.

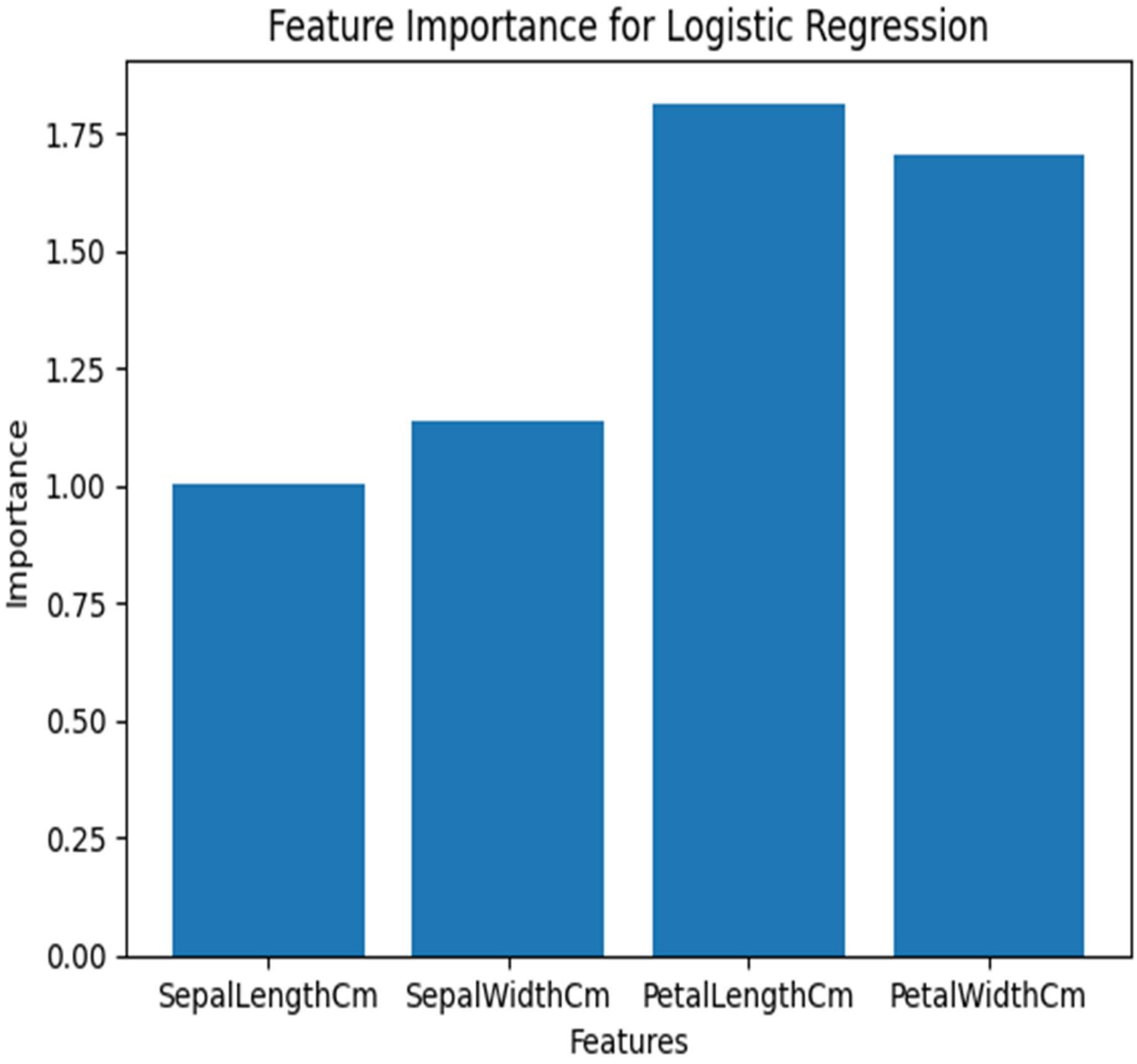


True label \ Predicted label	Iris-setosa	Iris-versicolor	Iris-virginica
Iris-setosa	10	0	0
Iris-versicolor	0	9	0
Iris-virginica	0	0	11

4. ROC Curve.


ROC Curve for Multiclass Classification
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve of class Iris-setosa (area = 1.00)
ROC curve of class Iris-versicolor (area = 0.99)
ROC curve of class Iris-virginica (area = 1.00)

5. Feature Importance Plot(Logistic Regression).


Feature Importance for Logistic Regression
Importance
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Features