02/27/2025

IRIS CLASSIFICATION WITH SUPPORT VECTOR MACHINE MODEL

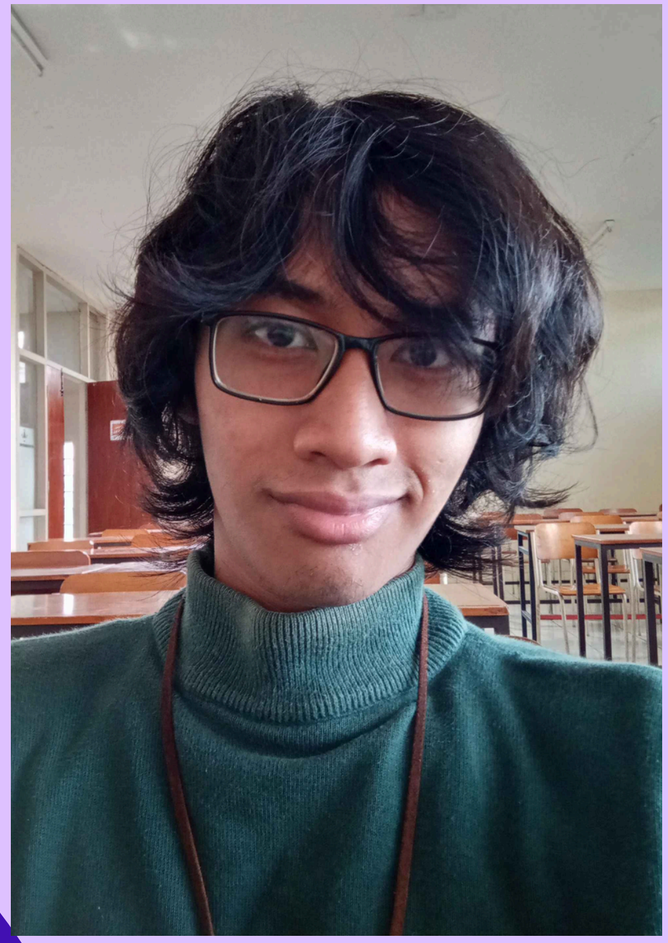
Created by : Muhammad Hasby As-shiddiqy

Iris Classification with Support Vector Machine Model

TABLE OF CONTENTS

By: Muhammad Hasby As-shiddiqy

ABOUT ME



MUHAMMAD HASBY AS-SHIDDIQY

Data Science | Data Analyst Enthusiast

Bandung, West Java, Indonesia

I’m an Undergraduate Computer Science student at Institut Teknologi Nasional (ITENAS). I’m deeply interested in Data Science dan Data Analytics.

ABOUT PROJECT

This project aims to classify Iris flower either as Setosa, Versicolor, or Virginica using one of the machine learning model such as Support Vector Machine (SVM).

The project uses Python as programming language and Jupyter Notebook or Google Colab as its code editor. The dataset used was the Iris dataset from Scikit-learn

By: Muhammad
Hasby As-shiddiqy

GOALS

- Build and train classification models using Support Vector Machine algorithm
- Evaluate the performance of the built model using accuracy as the main metrics
- Visualize the classification in any type of graphs so the there’s a clear view of the classification results

By: Muhammad Hasby As-shiddiqy

GOALS

- Build and train classification models using Support Vector Machine algorithm
- Evaluate the performance of the built model using accuracy as the main metrics
- Visualize the classification in any type of graphs so the there’s a clear view of the classification results

By: Muhammad Hasby As-shiddiqy

SUPPORT VECTOR MACHINE

Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression tasks. While it can handle regression problems, SVM is particularly well-suited for classification tasks.

SVM aims to find the optimal hyperplane in an N-dimensional space to separate data points into different classes. The algorithm maximizes the margin between the closest points of different classes.

Source:
geeksforgeeks.org/support-vector-machine-algorithm/

By: Muhammad Hasby As-shiddiqy

SOURCE CODE

1. Read Dataset

```
[24]  1 import pandas as pd # to manipulate data (DataFrame)
      2 from sklearn import datasets # import datasets from lib sklearn
      3
      4 # Load dataset Iris from sckit-learn
      5 iris = datasets.load_iris()
      6
      7 X = iris.data    # Input dataset for machine learning
      8 # Feature (sepal length, sepal width, petal length, petal width)
      9
     10 y = iris.target # Classification dataset for machine learning
     11 # Target Classes (Setosa, Versicolor, Virginica)
     12
     13 # Convert data features and target classes to DataFrame
     14 df_X = pd.DataFrame(X, columns=iris.feature_names)
     15 df_y = pd.Series(y, name='target')
```

```
[27]  1 # Merge Feature and Target Classes into one
      2 df_iris = pd.concat([df_X, df_y], axis=1)
      3
      4 df_iris
```


By: Muhammad Hasby As-shiddiqy

SOURCE CODE

2. Split Dataset

```
[31]  1 from sklearn.model_selection import train_test_split # to split data
      2
      3 # Split dataset to train and test the machine learning
      4 X_train, X_test, y_train, y_test = train_test_split(df_X, df_y, test_size=0.2, random_state=42, stratify=y) # Test 20%, Training 80%
```

```
[32]  1 # Verify if the classes are equally proportional
      2 # if it's equal, the dataset and model can be trained
      3 df_y.value_counts(normalize=True)
```

3. Dataset Normalization

```
[33]  1 from sklearn.preprocessing import StandardScaler # to normalize data
      2
      3 scaler = StandardScaler()
      4 X_train = scaler.fit_transform(X_train)
      5 X_test = scaler.transform(X_test)
```


By: Muhammad Hasby As-shiddiqy

SOURCE CODE

4. Choosing and Train the Machine Learning Model

```
1 from sklearn.svm import SVC # Model or Algorithm used for the machine learning (SVM)
2 #Support Vector Machine
3
4 # Making the SVM model with the kernel 'rbf' (Radial Basis Function)
5 svm_model = SVC(kernel='rbf', C=1.0, gamma='scale')
6
7 # Train the model
8 svm_model.fit(X_train, y_train)
```

5. Predict and Evaluate Model Performance

```
[35] 1 from sklearn.metrics import accuracy_score, classification_report, confusion_matrix # to evaluate machine learning model performance
     2
     3 y_pred = svm_model.predict(X_test)
     4
     5 # Evaluate
     6 accuracy = accuracy_score(y_test, y_pred)
     7 conf_matrix = confusion_matrix(y_test, y_pred)
     8 class_report = classification_report(y_test, y_pred)
```


By: Muhammad Hasby As-shiddiqy

SOURCE CODE

6. Evaluation Results

[36]
```python
# Print Results
print(f"Accuracy: {accuracy * 100:.2f}%")
print("\nConfusion Matrix:")
print(conf_matrix)

print("\nClassification:")
print(classification_report(y_test, y_pred, target_names=iris.target_names))
```


By: Muhammad
Hasby As-shiddiqy

SOURCE CODE

7. Visualize with Confusion Matrix

[37]
```python
import matplotlib.pyplot as plt # to visualize data in graph
import seaborn as sns # to colorize visualitzation

# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)

# Plot Confusion Matrix using seaborn
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
            xticklabels=iris.target_names, yticklabels=iris.target_names)
plt.xlabel("Prediction")
plt.ylabel("Actual")
plt.title("Confusion Matrix of Iris Dataset")
plt.show()
```


By: Muhammad Hasby As-shiddiqy

SOURCE CODE

8. Visualize with Pairplot

```
[38]  1 import matplotlib.pyplot as plt # to visualize data in graph
      2 import seaborn as sns # to colorize visualitzation
      3
      4 df_X["target"] = df_y.map({0: "Setosa", 1: "Versicolor", 2: "Virginica"})
      5
      6 sns.pairplot(df_X, hue="target", palette="coolwarm")
      7 plt.suptitle("Pairplot of Iris Dataset", y=1.02)
      8 plt.show()
      9
```


By: Muhammad
Hasby As-shiddiqy

SOURCE CODE

9. Visualize with Swarmplot

[39]

```python
import matplotlib.pyplot as plt # to visualize data in graph
import seaborn as sns # to colorize visualitzation

# Merge dataframe feature and target label
df = df_X.copy()
df["target"] = df_y.map({0: "Setosa", 1: "Versicolor", 2: "Virginica"})  # Menambahkan label target
df_melted = df.melt(id_vars='target', var_name='Feature', value_name='Value')

# Create subplot 1 row 3 column
fig, axes = plt.subplots(1, 3, figsize=(18, 6), sharey=True)

# List classes name
target_classes = ["Setosa", "Versicolor", "Virginica"]

# Loop to create 3 graph (1 for each classes)
for i, target in enumerate(target_classes):
    ax = axes[i]  # Pilih subplot ke-i
    sns.swarmplot(x='Feature', y='Value', data=df_melted[df_melted["target"] == target],
                  palette='deep', ax=ax)

    ax.set_title(f"{target}", fontsize=14)
    ax.set_xticklabels(ax.get_xticklabels(), rotation=45)  # Rotate the label so it will looks good
    ax.set_xlabel("Feature")
    ax.set_ylabel("Value" if i == 0 else "")  # Only on the first sublot that has label Y

    # Insert grid to every subplot
    ax.grid(True, linestyle='--', alpha=0.7)

plt.suptitle("Swarm Plot of Iris Dataset Features", fontsize=16)
plt.tight_layout()
plt.show()
```


By: Muhammad Hasby As-shiddiqy

SOURCE CODE

10. Visualize with Ditribution Target Class

[19]

```python
import matplotlib.pyplot as plt # to visualize data in graph
import seaborn as sns # to colorize visualitzation
import numpy as np # for array calculation


# Visualize the distribution of target classes
sns.countplot(x='target', data=df_y.to_frame(), palette="pastel")
plt.title('Distribution of Target Classes of Iris Dataset')
plt.xlabel('Class (0: Setosa, 1: Versicolor, 2: Virginica)')
plt.ylabel('Count')

# Show label every 5 number
max_count = df_y.value_counts().max()  # max value for Y
plt.yticks(np.arange(0, max_count + 5, 5))  # Show label every 5 number

# Show Horizontal Grid
plt.grid(axis='y', linestyle='--', alpha=0.7)

plt.show()
```


By: Muhammad Hasby As-shiddiqy

RESULTS

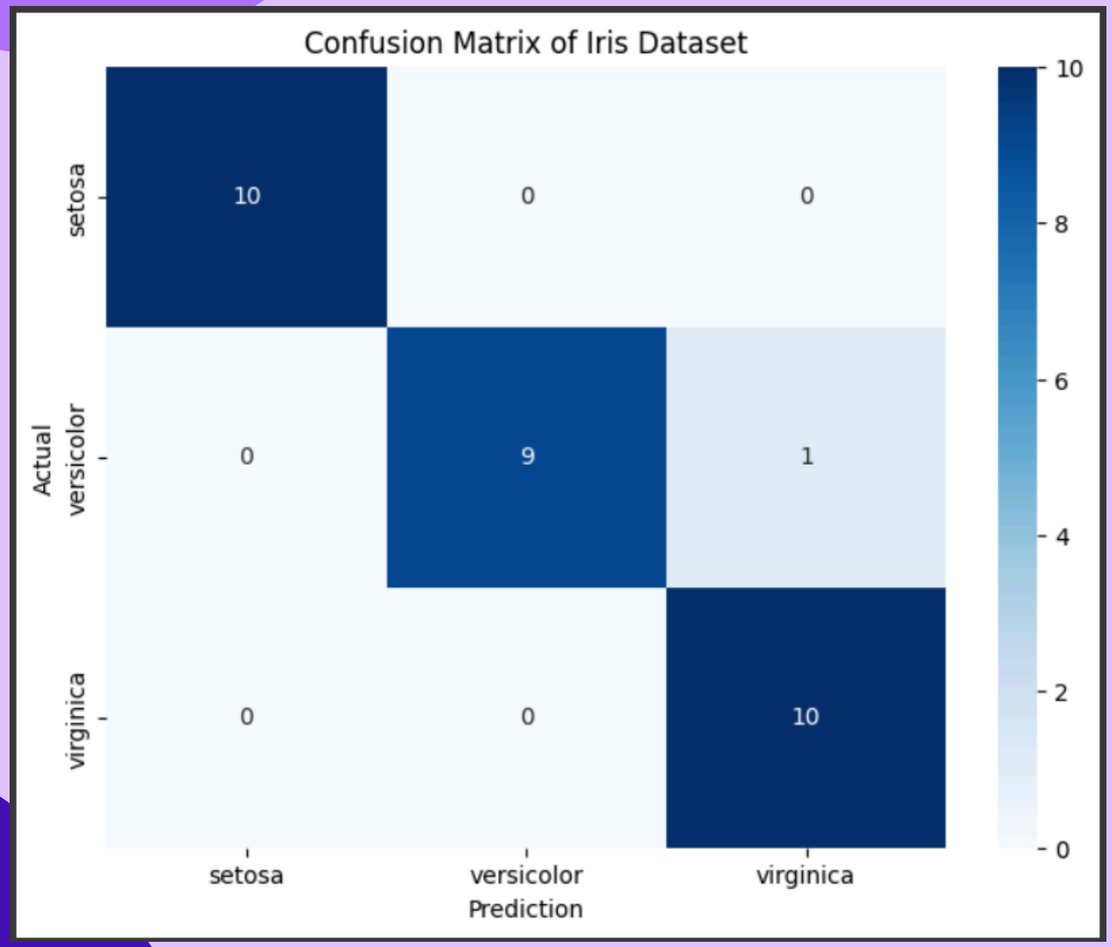
Evaluation Results

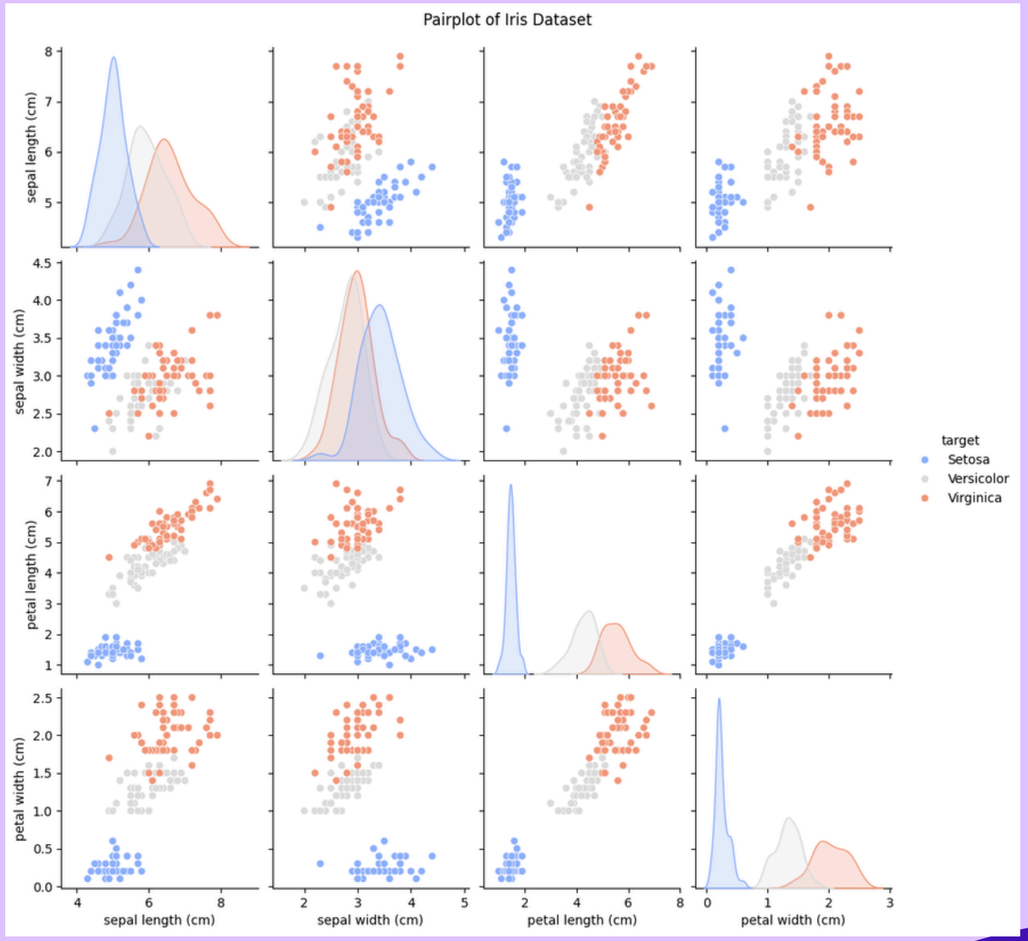
```
Accuracy: 96.67%

Confusion Matrix:
[[10  0  0]
 [ 0  9  1]
 [ 0  0 10]]

Classification:
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        10
  versicolor       1.00      0.90      0.95        10
   virginica       0.91      1.00      0.95        10

    accuracy                           0.97        30
   macro avg       0.97      0.97      0.97        30
weighted avg       0.97      0.97      0.97        30
```


By: Muhammad Hasby As-shiddiqy

RESULTS

Visualize with Confusion Matrix and Pairplot

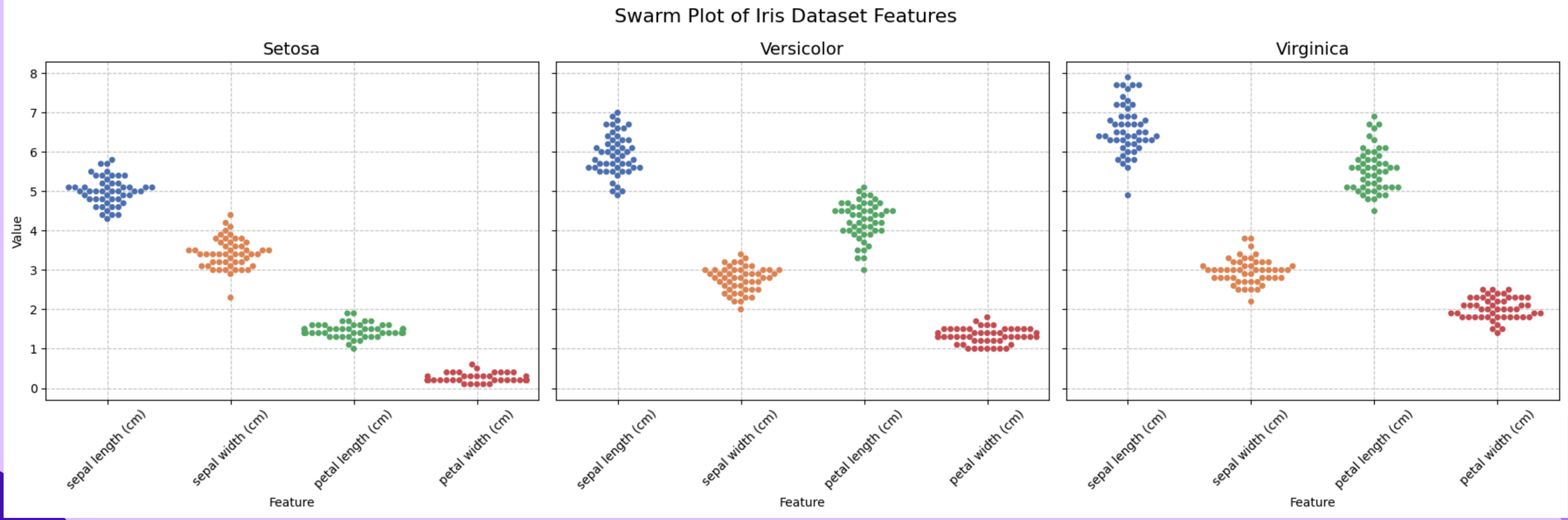




By: Muhammad
Hasby As-shiddiqy

RESULTS

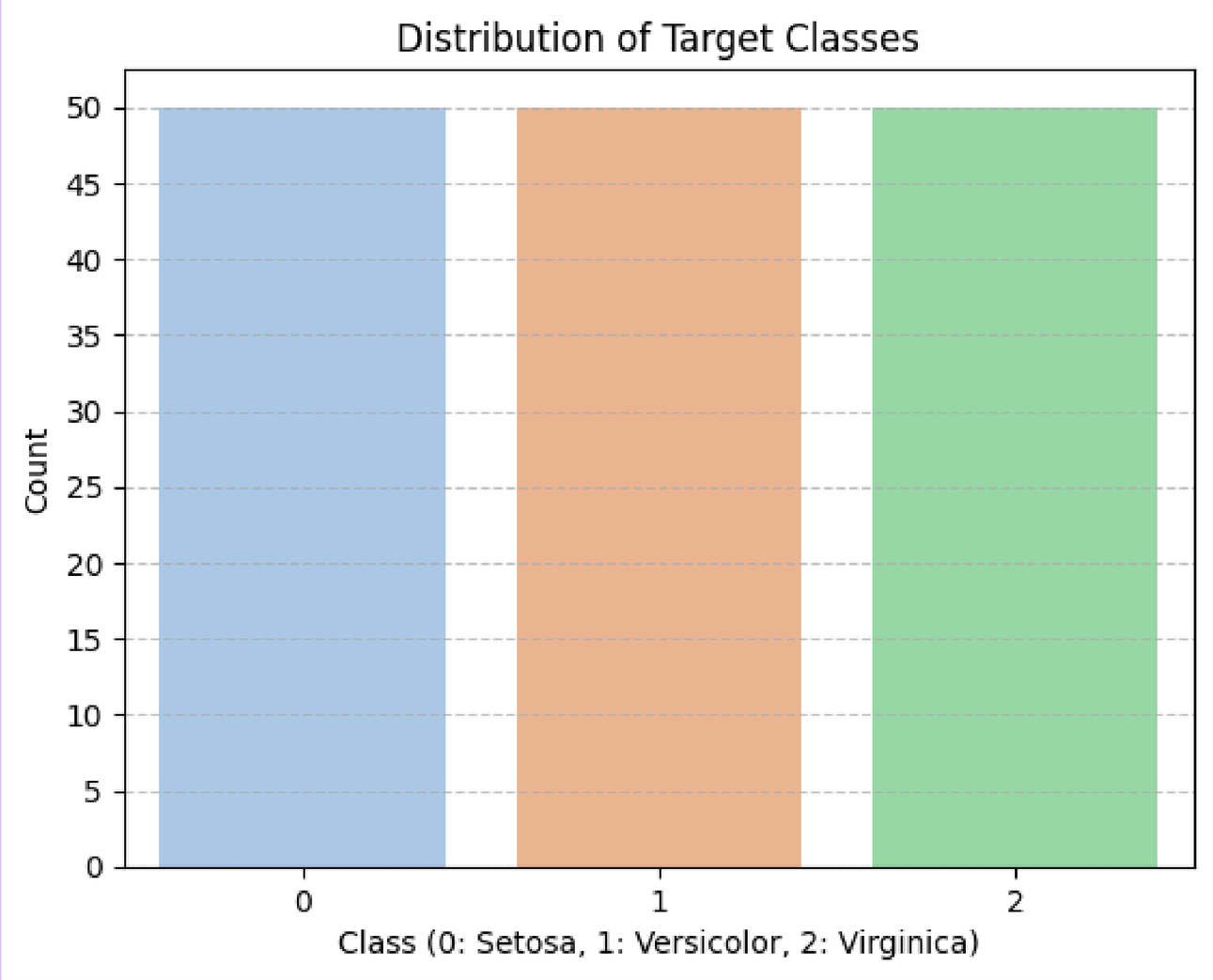
Visualize with Swarmplot



By: Muhammad
Hasby As-shiddiqy

RESULTS

Visualize with Distribution target Class



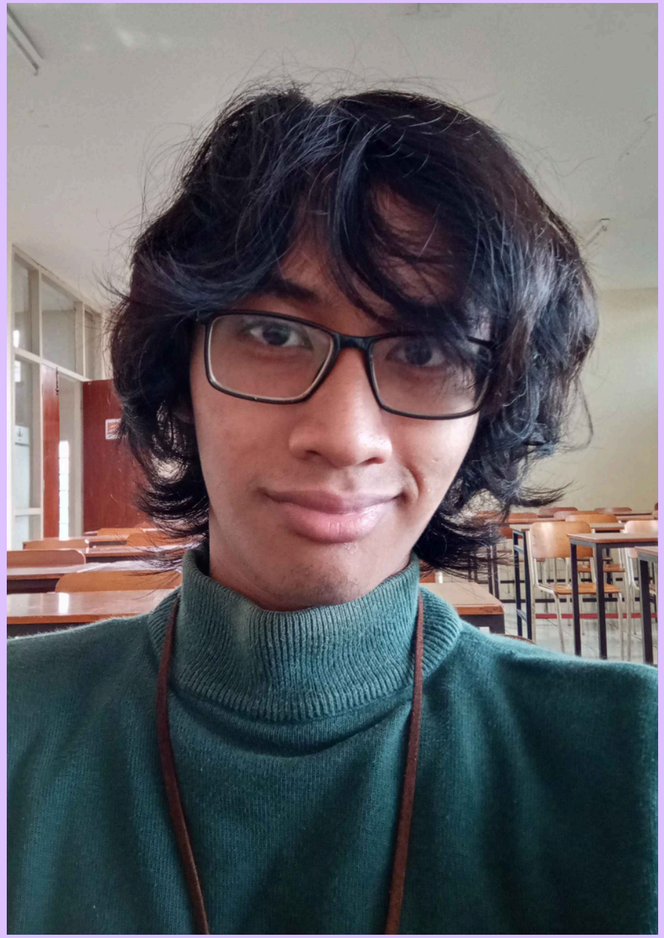
By: Muhammad
Hasby As-shiddiqy

CONTACT ME

Email by1frost.business@gmail.com

LinkedIn hasbyas1

Github hasbyas1



By: Muhammad
Hasby As-shiddiqy