**TITLE:** - Iris Flower Classification – Classify flower species based on petal and sepal dimensionsusing the Iris dataset.

**INTRODUCTION**  
  
  
**Objective:** The objective of this project is to build a machine learning model capable of classifying iris flowers into three species (Setosa, Versicolour, Virginica) based on their physical attributes: sepal length, sepal width, petal length, and petal width. The model aims to achieve high accuracy and generalization using Logistic Regression while also providing useful visualizations to understand the classification process.

**DATA SET DESCRIPTION:**

The Iris dataset is a well-known dataset introduced by Ronald Fisher in 1936. It contains 150 samples of iris flowers, equally distributed among three classes:

* Setosa (0)
* Versicolour (1)
* Virginica (2)

**Features:**

* Sepal Length (cm)
* Sepal Width (cm)
* Petal Length (cm)
* Petal Width (cm).

**METHODOLOGY**

**The code works in the following manner: -**

1. **Data Collection:** The Iris dataset was loaded using the load\_iris function from the Scikit-learn library. This dataset contains 150 samples, each with four features (sepal length, sepal width, petal length, and petal width) and a target variable indicating the species of the iris flower.
2. **Data Preprocessing:** Before training the model, the dataset was split into training and testing sets using the train\_test\_split function. This ensures that the model can be evaluated on unseen data. The training set consisted of 80% of the data, while the testing set consisted of the remaining 20%.
3. **Feature Standardization:** To ensure that the features are on a similar scale, standardization was performed using the StandardScaler class from Scikit-learn. This step is crucial for algorithms like KNN, which rely on distance calculations.
4. **Model Training:** The K-Nearest Neighbors (KNN) classifier was chosen for this classification task. The classifier was initialized with n\_neighbors set to 3. The model was then trained on the standardized training data using the fit method.
5. **Model Evaluation:** After training the model, predictions were made on the testing set using the predict method. The accuracy of the model was then calculated using the accuracy\_score function, which compares the predicted labels with the true labels of the testing set.

**CODE**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn import datasets

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

iris = datasets.load\_iris()

X = iris.data

y = iris.target

# Convert to DataFrame for visualization

iris\_df = pd.DataFrame(X, columns=iris.feature\_names)

iris\_df['species'] = iris.target

sns.pairplot(iris\_df, hue='species')

plt.show()

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

model = LogisticRegression(max\_iter=200)

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

report = classification\_report(y\_test, y\_pred, target\_names=iris.target\_names)

print(f"Accuracy: {accuracy \* 100:.2f}%")

print("\nClassification Report:\n", report)

cm = confusion\_matrix(y\_test, y\_pred)

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=iris.target\_names, yticklabels=iris.target\_names)

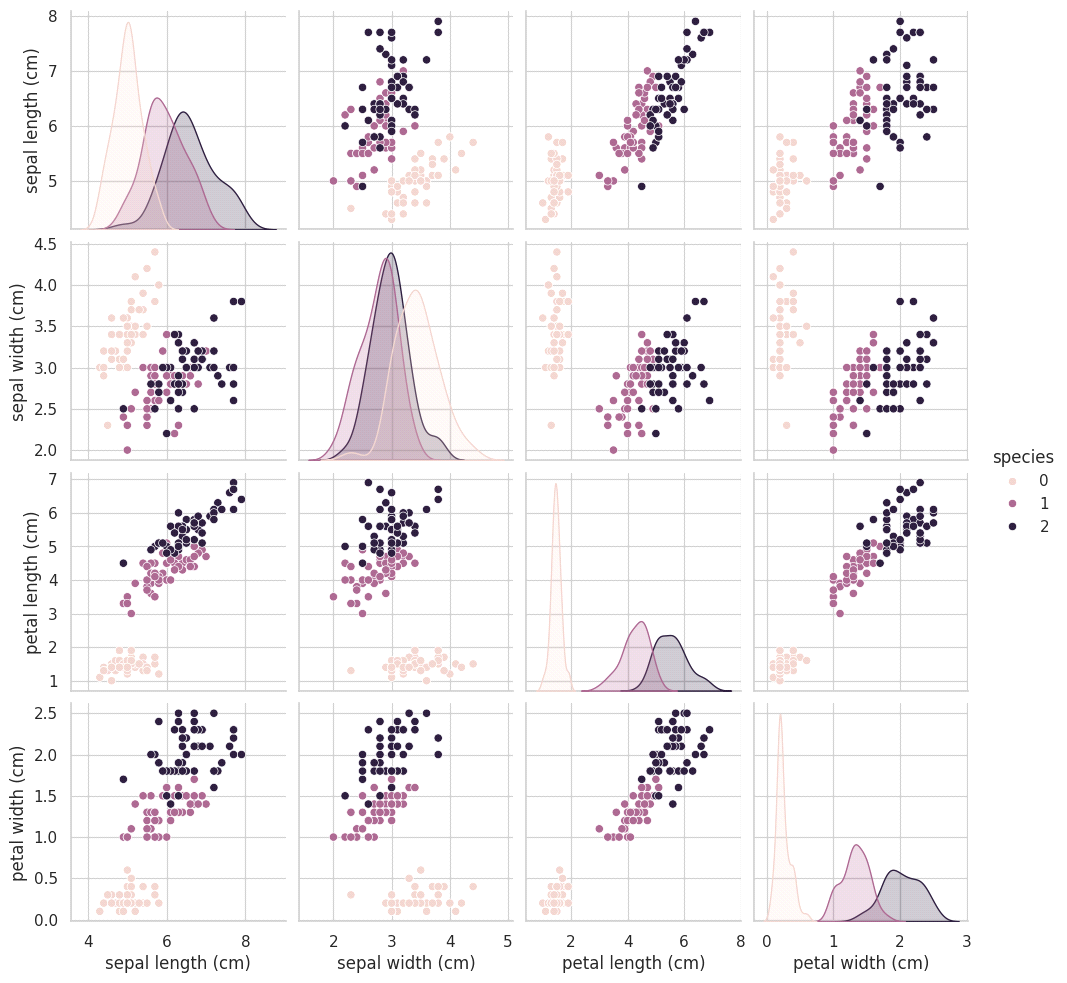
plt.title('Confusion Matrix')

plt.xlabel('Predicted Labels')

plt.ylabel('True Labels')

plt.show()

**OUTPUT**

Accuracy: 100.00%

Classification Report

precision recall f1-score support

setosa 1.00 1.00 1.00 10

versicolor 1.00 1.00 1.00 9

virginica 1.00 1.00 1.00 11

accuracy 1.00 30

macro avg 1.00 1.00 1.00 30

weighted avg 1.00 1.00 1.00 30

