# Import necessary libraries

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.datasets import load\_diabetes

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, auc

# Load the diabetes dataset

diabetes = load\_diabetes()

X, y = diabetes.data, diabetes.target

# Convert the target variable to binary (1 for diabetes, 0 for no diabetes)

y\_binary = (y > np.median(y)).astype(int)

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y\_binary, 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()

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

# Evaluate the model

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

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

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

output:

Accuracy: 73.03%

# evaluate the model

print("Confusion Matrix:\n", confusion\_matrix(y\_test, y\_pred))

print("\nClassification Report:\n", classification\_report(y\_test, y\_pred))

output:

Confusion Matrix:

[[36 13]

[11 29]]

Classification Report:

precision recall f1-score support

0 0.77 0.73 0.75 49

1 0.69 0.72 0.71 40

accuracy 0.73 89

macro avg 0.73 0.73 0.73 89

weighted avg 0.73 0.73 0.73 89

# Visualize the decision boundary with accuracy information

plt.figure(figsize=(8, 6))

sns.scatterplot(x=X\_test[:, 2], y=X\_test[:, 8], hue=y\_test, palette={

0: 'blue', 1: 'red'}, marker='o')

plt.xlabel("BMI")

plt.ylabel("Age")

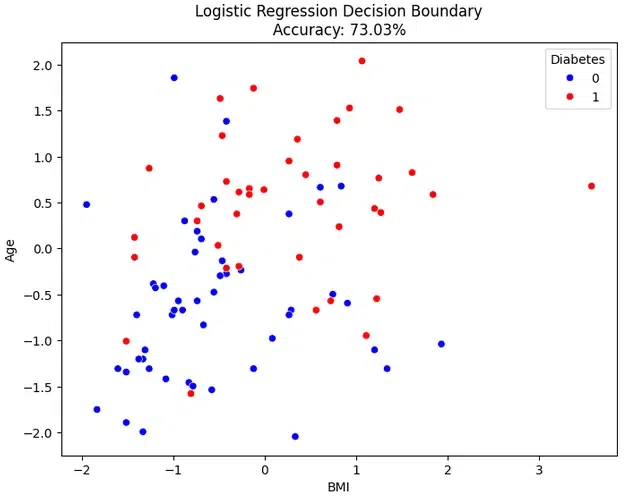
plt.title("Logistic Regression Decision Boundary\nAccuracy: {:.2f}%".format(

accuracy \* 100))

plt.legend(title="Diabetes", loc="upper right")

plt.show()

output:



# Plot ROC Curve

y\_prob = model.predict\_proba(X\_test)[:, 1]

fpr, tpr, thresholds = roc\_curve(y\_test, y\_prob)

roc\_auc = auc(fpr, tpr)

plt.figure(figsize=(8, 6))

plt.plot(fpr, tpr, color='darkorange', lw=2,

label=f'ROC Curve (AUC = {roc\_auc:.2f})')

plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--', label='Random')

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic (ROC) Curve\nAccuracy: {:.2f}%'.format(

accuracy \* 100))

plt.legend(loc="lower right")

plt.show()

output:

