''' ithe dataset internet varun ghetlay tr tumchya dataset anusaar columns che names dya code mdhe '''

import tensorflow as tf

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import accuracy\_score, classification\_report

from sklearn.datasets import load\_diabetes # Optional: Use another binary dataset

import pandas as pd

# Load a binary classification dataset

# We'll use Pima Indians Diabetes dataset from a CSV

url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"

column\_names = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',

'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome']

df = pd.read\_csv(url, names=column\_names)

# Split features and labels

X = df.drop('Outcome', axis=1).values

y = df['Outcome'].values

# Split into training and test sets

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

# Scale the data

scaler = StandardScaler()

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

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

# Logistic regression model (single dense layer with sigmoid)

logistic\_model = tf.keras.Sequential([

tf.keras.Input(shape=(X\_train.shape[1],)),

tf.keras.layers.Dense(1, activation='sigmoid')

])

logistic\_model.compile(optimizer='adam',

loss='binary\_crossentropy',

metrics=['accuracy'])

# Train the logistic regression model

logistic\_model.fit(X\_train, y\_train, epochs=100, verbose=0)

# Evaluate

loss, acc = logistic\_model.evaluate(X\_test, y\_test, verbose=0)

print(f"Logistic Regression Accuracy: {acc:.4f}")

# Predictions and classification report

y\_pred\_log = (logistic\_model.predict(X\_test) > 0.5).astype("int32")

print(classification\_report(y\_test, y\_pred\_log))