Exp 3

# Step 1: Install Required Libraries (if needed)

!pip install -q tensorflow scikit-learn pandas matplotlib

# Step 2: Import Libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

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

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import classification\_report, confusion\_matrix

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense

# Step 3: Load Dataset (PIMA Diabetes Dataset)

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

columns = ["Pregnancies", "Glucose", "BloodPressure", "SkinThickness", "Insulin",

           "BMI", "DiabetesPedigreeFunction", "Age", "Outcome"]

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

# Step 4: Data Exploration (Optional)

print(df.head())

print(df.describe())

print("Class Distribution:\n", df['Outcome'].value\_counts())

# Step 5: Data Preprocessing

X = df.drop("Outcome", axis=1)

y = df["Outcome"]

# Normalize features

scaler = StandardScaler()

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

# Train-Test Split

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

# Step 6: Build the ANN Model

model = Sequential()

model.add(Dense(16, input\_dim=X\_train.shape[1], activation='relu'))

model.add(Dense(8, activation='relu'))

model.add(Dense(1, activation='sigmoid'))  # Binary classification

# Step 7: Compile the Model

model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy'])

model.summary()   # to see the model

# Step 8: Train the Model

history = model.fit(X\_train, y\_train, epochs=100, batch\_size=16, validation\_split=0.1, verbose=0)

# Step 9: Evaluate the Model

loss, accuracy = model.evaluate(X\_test, y\_test)

print(f"\nTest Accuracy: {accuracy:.4f}")

# Step 10: Predict & Report

y\_pred = (model.predict(X\_test) > 0.5).astype("int32")

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

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

# Step 11: Plot Accuracy & Loss

plt.figure(figsize=(12, 5))

# Accuracy Plot

plt.subplot(1, 2, 1)

plt.plot(history.history['accuracy'], label='Train Acc')

plt.plot(history.history['val\_accuracy'], label='Val Acc')

plt.title('Model Accuracy')

plt.xlabel('Epoch')

plt.ylabel('Accuracy')

plt.legend()

# Loss Plot

plt.subplot(1, 2, 2)

plt.plot(history.history['loss'], label='Train Loss')

plt.plot(history.history['val\_loss'], label='Val Loss')

plt.title('Model Loss')

plt.xlabel('Epoch')

plt.ylabel('Loss')

plt.legend()

plt.tight\_layout()

plt.show()









