import pandas as pd

import numpy as np

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

from sklearn.metrics import accuracy\_score

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, Flatten, Dense

from xgboost import XGBClassifier

# Load WBCD dataset

url = "https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.data"

columns = [

"ID", "Diagnosis", # First two columns

# Remaining 30 features

"Radius\_Mean", "Texture\_Mean", "Perimeter\_Mean", "Area\_Mean", "Smoothness\_Mean",

"Compactness\_Mean", "Concavity\_Mean", "Concave\_Points\_Mean", "Symmetry\_Mean", "Fractal\_Dimension\_Mean",

"Radius\_SE", "Texture\_SE", "Perimeter\_SE", "Area\_SE", "Smoothness\_SE",

"Compactness\_SE", "Concavity\_SE", "Concave\_Points\_SE", "Symmetry\_SE", "Fractal\_Dimension\_SE",

"Radius\_Worst", "Texture\_Worst", "Perimeter\_Worst", "Area\_Worst", "Smoothness\_Worst",

"Compactness\_Worst", "Concavity\_Worst", "Concave\_Points\_Worst", "Symmetry\_Worst", "Fractal\_Dimension\_Worst"

]

data = pd.read\_csv(url, header=None, names=columns)

# Preprocess data

X = data.iloc[:, 2:].values # Features (30 numeric columns)

y = (data.iloc[:, 1] == "M").astype(int).values # Labels: 1 for Malignant, 0 for Benign

# Normalize features

X = X / X.max(axis=0)

# Reshape for CNN (e.g., 30 features -> 5x6 grid)

X\_cnn = X.reshape(X.shape[0], 5, 6, 1) # Reshaped to 5x6 "image" with 1 channel

# Train-test split

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

# Build CNN model

cnn = Sequential([

Conv2D(32, (3, 3), activation='relu', input\_shape=(5, 6, 1)),

Flatten(),

Dense(128, activation='relu'),

Dense(64, activation='relu'),

Dense(16, activation='relu') # Feature extraction layer

])

# Compile CNN

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

# Train CNN

cnn.fit(X\_train, y\_train, epochs=10, batch\_size=16, verbose=1)

# Extract features from CNN

cnn\_feature\_extractor = Sequential(cnn.layers[:-1]) # Exclude final dense layer

X\_train\_features = cnn\_feature\_extractor.predict(X\_train)

X\_test\_features = cnn\_feature\_extractor.predict(X\_test)

# Train XGBoost on CNN features

xgb = XGBClassifier()

xgb.fit(X\_train\_features, y\_train)

# Predict with XGBoost

y\_pred = xgb.predict(X\_test\_features)

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

print(f"Accuracy of CNN+XGB model: {accuracy \* 100:.2f}%")