```
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 tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.optimizers import Adam

# === Load datasets ===
rayleigh_df = pd.read_csv("DNN_RayleighTrainingData_FixedTheta.csv")
correlation_mae_df = pd.read_csv("MAE_vs_G.csv")

# === Extract features and labels ===
X = rayleigh_df.filter(like="Iter").values
y = rayleigh_df["True_Angle"].values
nlos_gain = rayleigh_df["NLOS_Gain"].values

# === Normalize ===
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# === Split ===
X_train, X_test, y_train, y_test, _, gain_test = train_test_split(
    X_scaled, y, nlos_gain, test_size=0.2, stratify=y, random_state=42
)

# === Define and train model ===
model = Sequential([
    Dense(100, activation='relu', input_shape=(X_train.shape[1],)),
    Dropout(0.2),
    Dense(80, activation='relu'),
    Dropout(0.2),
    Dense(60, activation='relu'),
    Dropout(0.2),
    Dense(1, activation='linear')
])
model.compile(optimizer=Adam(learning_rate=0.001),
              loss='mean_squared_error',
              metrics=['mae'])

model.fit(X_train, y_train, epochs=50, batch_size=32, validation_split=0.1, verbose=1)

# === Predictions & MAE by NLOS Gain ===
y_pred = model.predict(X_test).flatten()
abs_err = np.abs(y_pred - y_test)
unique_gains = np.sort(np.unique(gain_test))
mae_per_gain = [np.mean(abs_err[gain_test == g]) for g in unique_gains]

# === Reference MAE data ===
ref_gains = correlation_mae_df["NLOS_Gain"]
ref_mae = correlation_mae_df["Mean_Absolute_Error"]

# === Plot: MAE vs NLOS Gain ===
plt.figure(figsize=(12, 6))
x = np.arange(len(unique_gains))
width = 0.25

plt.bar(x - width, ref_mae[:len(x)], width, label="Correlation MAE", color='orange')
plt.bar(x, mae_per_gain, width, label="DNN Validation MAE", color='blue')

plt.xticks(x, [str(g) for g in unique_gains])
plt.xlabel("NLOS Gain")
plt.ylabel("Mean Absolute Error (°)")
plt.title("MAE vs NLOS Gain (Rayleigh Model)")
plt.legend()
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()

# === Scatter Plot: Predicted vs Actual Angles ===
plt.figure(figsize=(8, 8))
plt.scatter(y_test, y_pred, alpha=0.6, edgecolor='k')
plt.plot([0, 90], [0, 90], 'r--', label="Ideal Prediction")
plt.xlabel("True Angle (°)")
plt.ylabel("Predicted Angle (°)")
plt.title("Predicted vs Actual Angles (Rayleigh Model)")
plt.legend()
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

Show hidden output