NeuralNet

November 9, 2024

[20]:
```
try:
    import os
    import glob
    import numpy as np
    import pandas as pd

    from sklearn.model_selection import train_test_split
    from sklearn.preprocessing import StandardScaler, OneHotEncoder
    from sklearn.compose import ColumnTransformer
    from sklearn.pipeline import Pipeline

except Exception as e:
    print(f"Error : {e}")
```

[21]:
```
# Find the CSV file in the Datasets directory
data_path = '../Datasets/*.csv'
file_list = glob.glob(data_path)

for file in file_list:
    print(f"Found file: {file}")

# Ensure there is exactly one file
if len(file_list) == 1:
    # Load the dataset
    df = pd.read_csv(file_list[0])
    print(f"Loaded dataset: {file_list[0]}")
else:
    raise FileNotFoundError("No CSV file found or multiple CSV files found in␣
 ↪the Datasets directory.")
```

```
Found file: ../Datasets/Dataset.csv
Loaded dataset: ../Datasets/Dataset.csv
```

[22]:
```
# File path to save the trained model
destination = '../Models/'
os.makedirs(destination, exist_ok=True)
print(f"Model will be saved to: {destination}")
```

```
Model will be saved to: ../Models/
```

[23]:
```
# Features (X) and target (y)
X = df.drop(columns=['Lifespan'])  # Features excluding the target variable
y = df['Lifespan']  # Target variable

# Define categorical and numerical features
categorical_features = X.select_dtypes(include=['object', 'category']).columns.
 ↪tolist()
numerical_features = X.select_dtypes(include=['int64', 'float64']).columns.
 ↪tolist()

# Preprocessing for numerical features: Standard Scaling
numerical_transformer = StandardScaler()

# Preprocessing for categorical features: One-Hot Encoding
categorical_transformer = OneHotEncoder(drop='first', handle_unknown='ignore')

# Create a ColumnTransformer to apply transformations
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_features),
        ('cat', categorical_transformer, categorical_features)
    ]
)

# Create a preprocessing pipeline
pipeline = Pipeline(steps=[('preprocessor', preprocessor)])

# Fit and transform the data
X_processed = pipeline.fit_transform(X)

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_processed, y, test_size=0.
 ↪2, random_state=42)

# Checking the shapes of the processed data
print(f"Training features shape: {X_train.shape}")
print(f"Testing features shape: {X_test.shape}")
```

```
Training features shape: (800, 19)
Testing features shape: (200, 19)
```

[24]:
```
# Import necessary libraries for Neural Network
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Input
from tensorflow.keras.optimizers import Adam
```

```
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error,␣
 ↪mean_squared_log_error
import numpy as np

# Define the model
model = Sequential()
model.add(Input(shape=(X_train.shape[1],)))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.2))  # Optional dropout to avoid overfitting
model.add(Dense(32, activation='relu'))
model.add(Dense(1, activation='linear'))  # Using a linear activation for␣
 ↪regression

# Compile the model
model.compile(optimizer=Adam(learning_rate=0.001), loss='mean_squared_error',␣
 ↪metrics=['mae'])

# Train the model
history = model.fit(X_train, y_train, validation_split=0.2, epochs=50,␣
 ↪batch_size=32)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the performance
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
msle = mean_squared_log_error(y_test, y_pred)

print(f"Neural Network RMSE: {rmse:.2f}")
print(f"Neural Network R² Score: {r2:.2f}")
print(f"Neural Network MAE: {mae:.2f}")
print(f"Neural Network MSLE: {msle:.2f}")

# Optional: Plot the loss during training
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss (MSE)')
plt.title('Training and Validation Loss Over Epochs')
plt.legend()
plt.show()
```

```
Epoch 1/50
20/20            0s 3ms/step - loss:
1814200.8750 - mae: 1302.2971 - val_loss: 1852780.3750 - val_mae: 1320.0710
Epoch 2/50
20/20            0s 942us/step -
loss: 1823857.8750 - mae: 1305.3876 - val_loss: 1847886.7500 - val_mae:
1318.1725
Epoch 3/50
20/20            0s 902us/step -
loss: 1777887.2500 - mae: 1289.7183 - val_loss: 1839104.6250 - val_mae:
1314.7776
Epoch 4/50
20/20            0s 932us/step -
loss: 1772426.3750 - mae: 1285.2733 - val_loss: 1823795.2500 - val_mae:
1308.8718
Epoch 5/50
20/20            0s 924us/step -
loss: 1794246.3750 - mae: 1293.3633 - val_loss: 1798684.0000 - val_mae:
1299.1725
Epoch 6/50
20/20            0s 905us/step -
loss: 1769273.8750 - mae: 1281.4574 - val_loss: 1760334.7500 - val_mae:
1284.2705
Epoch 7/50
20/20            0s 842us/step -
loss: 1714897.3750 - mae: 1258.4670 - val_loss: 1704971.2500 - val_mae:
1262.5103
Epoch 8/50
20/20            0s 978us/step -
loss: 1669304.7500 - mae: 1241.4894 - val_loss: 1629268.7500 - val_mae:
1232.2113
Epoch 9/50
20/20            0s 803us/step -
loss: 1566961.2500 - mae: 1199.8572 - val_loss: 1531523.7500 - val_mae:
1191.9856
Epoch 10/50
20/20            0s 821us/step -
loss: 1414400.8750 - mae: 1132.0464 - val_loss: 1411134.1250 - val_mae:
1140.5042
Epoch 11/50
20/20            0s 960us/step -
loss: 1338026.8750 - mae: 1104.6954 - val_loss: 1269512.6250 - val_mae:
1076.8298
Epoch 12/50
20/20            0s 839us/step -
loss: 1216499.3750 - mae: 1047.3787 - val_loss: 1112017.3750 - val_mae:
1001.2021
Epoch 13/50
```

```
20/20            0s 818us/step -
loss: 1015068.5000 - mae: 938.6147 - val_loss: 944890.8125 - val_mae: 913.8592
Epoch 14/50
20/20            0s 872us/step -
loss: 903364.7500 - mae: 878.5349 - val_loss: 776069.8750 - val_mae: 815.8257
Epoch 15/50
20/20            0s 1ms/step - loss:
683391.7500 - mae: 747.1827 - val_loss: 620359.2500 - val_mae: 712.5651
Epoch 16/50
20/20            0s 856us/step -
loss: 586229.2500 - mae: 673.8903 - val_loss: 482304.1875 - val_mae: 608.7082
Epoch 17/50
20/20            0s 1ms/step - loss:
451550.6250 - mae: 569.5538 - val_loss: 373402.3750 - val_mae: 517.3489
Epoch 18/50
20/20            0s 866us/step -
loss: 346573.1562 - mae: 489.9489 - val_loss: 290703.5625 - val_mae: 440.6732
Epoch 19/50
20/20            0s 2ms/step - loss:
268695.4375 - mae: 422.7924 - val_loss: 236018.5000 - val_mae: 387.2460
Epoch 20/50
20/20            0s 881us/step -
loss: 262141.9688 - mae: 411.2223 - val_loss: 202124.5312 - val_mae: 355.8801
Epoch 21/50
20/20            0s 871us/step -
loss: 229890.1406 - mae: 388.0386 - val_loss: 181945.2344 - val_mae: 339.2639
Epoch 22/50
20/20            0s 844us/step -
loss: 216549.0938 - mae: 387.1922 - val_loss: 170188.3281 - val_mae: 329.0403
Epoch 23/50
20/20            0s 990us/step -
loss: 204718.4844 - mae: 362.8419 - val_loss: 163970.4062 - val_mae: 322.7397
Epoch 24/50
20/20            0s 813us/step -
loss: 203259.6562 - mae: 363.8215 - val_loss: 160635.2969 - val_mae: 320.2120
Epoch 25/50
20/20            0s 842us/step -
loss: 188961.0156 - mae: 354.4363 - val_loss: 158058.5625 - val_mae: 318.4193
Epoch 26/50
20/20            0s 822us/step -
loss: 216446.8594 - mae: 377.3097 - val_loss: 156109.4688 - val_mae: 317.0804
Epoch 27/50
20/20            0s 870us/step -
loss: 202811.9844 - mae: 368.3314 - val_loss: 154923.7188 - val_mae: 316.3609
Epoch 28/50
20/20            0s 820us/step -
loss: 193252.2344 - mae: 361.6443 - val_loss: 154166.0469 - val_mae: 316.0194
Epoch 29/50
```

```
20/20            0s 862us/step -
loss: 188142.9219 - mae: 355.0952 - val_loss: 152889.3438 - val_mae: 314.8110
Epoch 30/50
20/20            0s 1ms/step - loss:
200776.9219 - mae: 372.1741 - val_loss: 152032.0625 - val_mae: 314.5005
Epoch 31/50
20/20            0s 849us/step -
loss: 187363.5781 - mae: 356.1980 - val_loss: 150575.9531 - val_mae: 313.0618
Epoch 32/50
20/20            0s 866us/step -
loss: 188659.4375 - mae: 357.7792 - val_loss: 149628.1250 - val_mae: 312.5059
Epoch 33/50
20/20            0s 905us/step -
loss: 171967.1719 - mae: 334.4113 - val_loss: 148515.8906 - val_mae: 311.5574
Epoch 34/50
20/20            0s 815us/step -
loss: 184809.6719 - mae: 349.9435 - val_loss: 147924.4219 - val_mae: 311.1056
Epoch 35/50
20/20            0s 2ms/step - loss:
193715.4219 - mae: 357.2389 - val_loss: 147406.5312 - val_mae: 310.8054
Epoch 36/50
20/20            0s 899us/step -
loss: 200428.7812 - mae: 365.8511 - val_loss: 146686.1562 - val_mae: 310.3503
Epoch 37/50
20/20            0s 831us/step -
loss: 180547.4531 - mae: 345.5062 - val_loss: 146345.1406 - val_mae: 310.1881
Epoch 38/50
20/20            0s 807us/step -
loss: 180323.2969 - mae: 339.6756 - val_loss: 145950.0625 - val_mae: 310.0602
Epoch 39/50
20/20            0s 820us/step -
loss: 166864.8438 - mae: 329.8791 - val_loss: 145309.9219 - val_mae: 309.2965
Epoch 40/50
20/20            0s 872us/step -
loss: 177611.6719 - mae: 348.4703 - val_loss: 143875.0781 - val_mae: 308.0212
Epoch 41/50
20/20            0s 838us/step -
loss: 177131.6250 - mae: 340.0251 - val_loss: 143091.9844 - val_mae: 307.2733
Epoch 42/50
20/20            0s 875us/step -
loss: 181801.9688 - mae: 349.6869 - val_loss: 142269.2344 - val_mae: 306.4320
Epoch 43/50
20/20            0s 862us/step -
loss: 182417.0938 - mae: 345.7777 - val_loss: 141818.8438 - val_mae: 306.0219
Epoch 44/50
20/20            0s 839us/step -
loss: 176925.8125 - mae: 340.8438 - val_loss: 141002.6562 - val_mae: 305.3901
Epoch 45/50
```

```
20/20             0s 833us/step -
loss: 171733.8594 - mae: 342.1100 - val_loss: 140528.0312 - val_mae: 304.6649
Epoch 46/50
20/20             0s 851us/step -
loss: 177510.2656 - mae: 345.2181 - val_loss: 139165.6250 - val_mae: 303.3529
Epoch 47/50
20/20             0s 800us/step -
loss: 169255.0312 - mae: 331.6056 - val_loss: 138875.8438 - val_mae: 302.8742
Epoch 48/50
20/20             0s 818us/step -
loss: 170596.3594 - mae: 337.3381 - val_loss: 138609.7812 - val_mae: 302.7893
Epoch 49/50
20/20             0s 836us/step -
loss: 167319.2344 - mae: 336.3355 - val_loss: 137213.2812 - val_mae: 301.4688
Epoch 50/50
20/20             0s 1ms/step - loss:
161878.8281 - mae: 335.6810 - val_loss: 136279.4062 - val_mae: 300.5347
7/7             0s 2ms/step
Neural Network RMSE: 386.39
Neural Network R² Score: -0.44
Neural Network MAE: 320.94
Neural Network MSLE: 0.10
```