```python
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```python
cd '/content/drive/MyDrive/Colab Notebooks/'
```

```
/content/drive/MyDrive/Colab Notebooks
```

```python
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from tensorflow.keras.utils import to_categorical
import numpy as np
!pip install missingno
!pip install plotly.express
import missingno as msno
import matplotlib.pyplot as plt
import plotly.express as px
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from keras.utils import plot_model
from IPython.display import Image
from sklearn.metrics import classification_report, accuracy_score

from keras_tuner.engine.hyperparameters import HyperParameters
from keras_tuner import RandomSearch, Hyperband, BayesianOptimization
from tensorflow.keras.utils import plot_model
from tensorflow.keras.callbacks import LearningRateScheduler, TensorBoard
import keras_tuner as kt
from sklearn.metrics import classification_report, confusion_matrix

```

```
Requirement already satisfied: missingno in /usr/local/lib/python3.10/dist-packages (0.5.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from missingno) (1.26.4)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from missingno) (3.7.1)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from missingno) (1.13.1)
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (from missingno) (0.13.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (1.3.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (4.53.1)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (1.4.7)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (24.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (3.1.4)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (2.8.2)
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.10/dist-packages (from seaborn->missingno) (2.1.4)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2->seaborn->missingno) (2024.2)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2->seaborn->missingno) (2024.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib->missingno) (1
Requirement already satisfied: plotly.express in /usr/local/lib/python3.10/dist-packages (0.4.1)
Requirement already satisfied: pandas>=0.20.0 in /usr/local/lib/python3.10/dist-packages (from plotly.express) (2.1.4)
Requirement already satisfied: plotly>=4.1.0 in /usr/local/lib/python3.10/dist-packages (from plotly.express) (5.24.1)
Requirement already satisfied: statsmodels>=0.9.0 in /usr/local/lib/python3.10/dist-packages (from plotly.express) (0.14.3)
Requirement already satisfied: scipy>=0.18 in /usr/local/lib/python3.10/dist-packages (from plotly.express) (1.13.1)
Requirement already satisfied: patsy>=0.5 in /usr/local/lib/python3.10/dist-packages (from plotly.express) (0.5.6)
Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.10/dist-packages (from plotly.express) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.20.0->plotly.express) (
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.20.0->plotly.express) (2024.2)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.20.0->plotly.express) (2024.1)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from patsy>=0.5->plotly.express) (1.16.0)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly>=4.1.0->plotly.express) (9.0.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from plotly>=4.1.0->plotly.express) (24.1)
```

```python
# Define the paths for train and test datasets in your Google Drive
train_path = '/content/drive/MyDrive/Colab Notebooks/train.csv'
test_path = '/content/drive/MyDrive/Colab Notebooks/test.csv'

# Load the datasets
import pandas as pd
train_df = pd.read_csv(train_path)
test_df = pd.read_csv(test_path)
```

```
# Check for missing values in train and test data
print(train_df.isnull().sum())

```

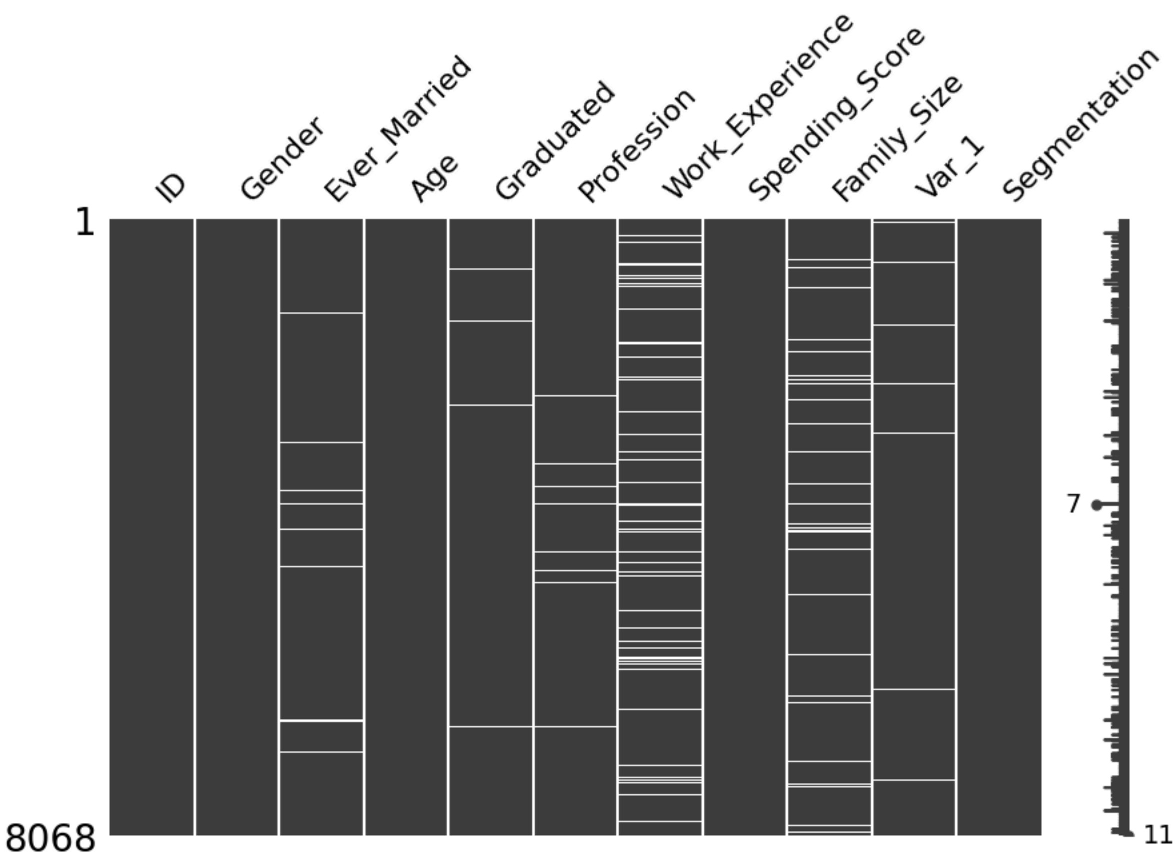
```
ID                   0
Gender               0
Ever_Married       140
Age                  0
Graduated           78
Profession         124
Work_Experience    829
Spending_Score       0
Family_Size        335
Var_1               76
Segmentation         0
dtype: int64
```

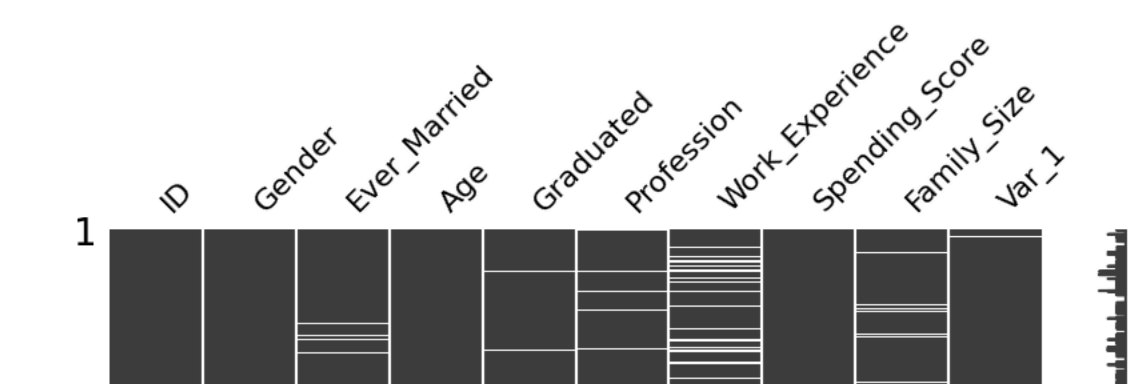
```
print(test_df.isnull().sum())
```

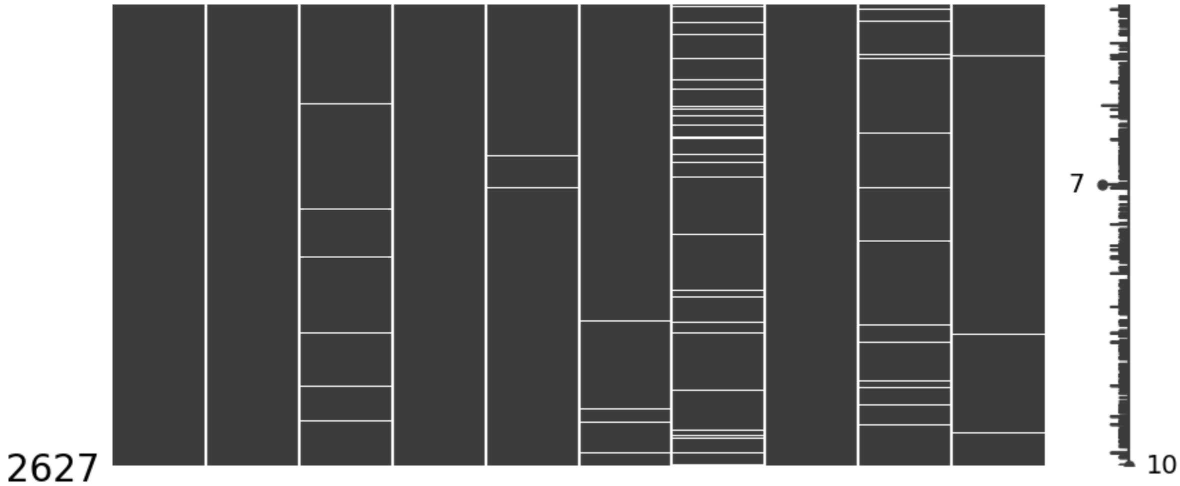
```
ID                   0
Gender               0
Ever_Married        50
Age                  0
Graduated           24
Profession          38
Work_Experience    269
Spending_Score       0
Family_Size        113
Var_1               32
dtype: int64
```

```
!pip install missingno
!pip install plotly.express
import missingno as msno
import matplotlib.pyplot as plt
import plotly.express as px
# Visualize missing values
msno.matrix(train_df, figsize=(10, 6)) #removed extra indentation
plt.show() #removed extra indentation
msno.matrix(test_df, figsize=(10, 6)) #removed extra indentation
plt.show() #removed extra indentation

# Plot distributions of numerical columns
for column in train_df.select_dtypes(include=['float64', 'int64']).columns:
    fig = px.histogram(train_df, x=column, title=f'Distribution of {column} in Train data')
    fig.show()
for column in test_df.select_dtypes(include=['float64', 'int64']).columns:
    fig = px.histogram(test_df, x=column, title=f'Distribution of {column} in Test data')
    fig.show()
```

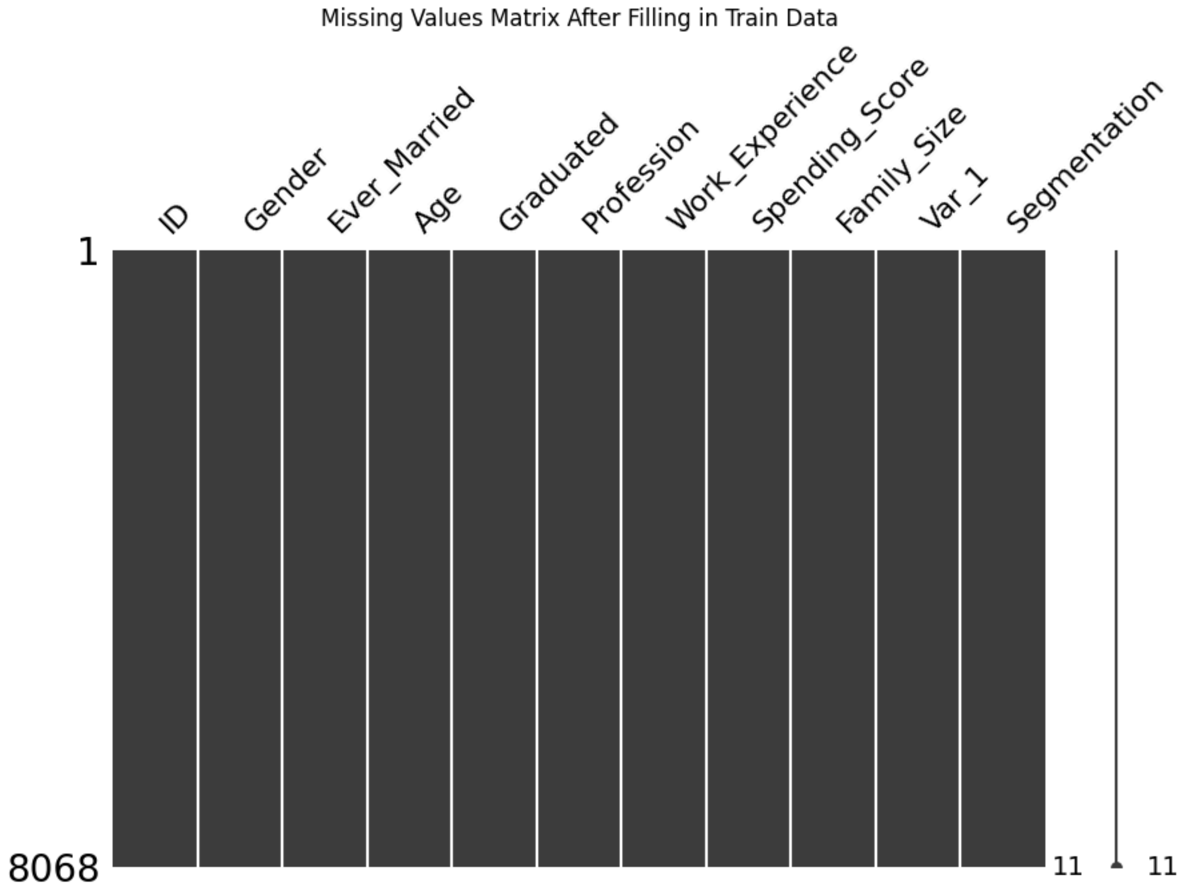
```
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from missingno) (1.26.4)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from missingno) (3.7.1)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from missingno) (1.13.1)
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (from missingno) (0.13.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (1.3.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (4.53.1)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (1.4.7)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (24.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (3.1.4)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (2.8.2)
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.10/dist-packages (from seaborn->missingno) (2.1.4)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2->seaborn->missingno) (2024.2
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2->seaborn->missingno) (2024
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib->missingno)
Requirement already satisfied: plotly.express in /usr/local/lib/python3.10/dist-packages (0.4.1)
Requirement already satisfied: pandas>=0.20.0 in /usr/local/lib/python3.10/dist-packages (from plotly.express) (2.1.4)
Requirement already satisfied: plotly>=4.1.0 in /usr/local/lib/python3.10/dist-packages (from plotly.express) (5.24.1)
Requirement already satisfied: statsmodels>=0.9.0 in /usr/local/lib/python3.10/dist-packages (from plotly.express) (0.14.3)
Requirement already satisfied: scipy>=0.18 in /usr/local/lib/python3.10/dist-packages (from plotly.express) (1.13.1)
Requirement already satisfied: patsy>=0.5 in /usr/local/lib/python3.10/dist-packages (from plotly.express) (0.5.6)
Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.10/dist-packages (from plotly.express) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.20.0->plotly.express
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.20.0->plotly.express) (2024.2)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.20.0->plotly.express) (2024.
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from patsy>=0.5->plotly.express) (1.16.0)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly>=4.1.0->plotly.express) (9.0.0
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from plotly>=4.1.0->plotly.express) (24.1)
```

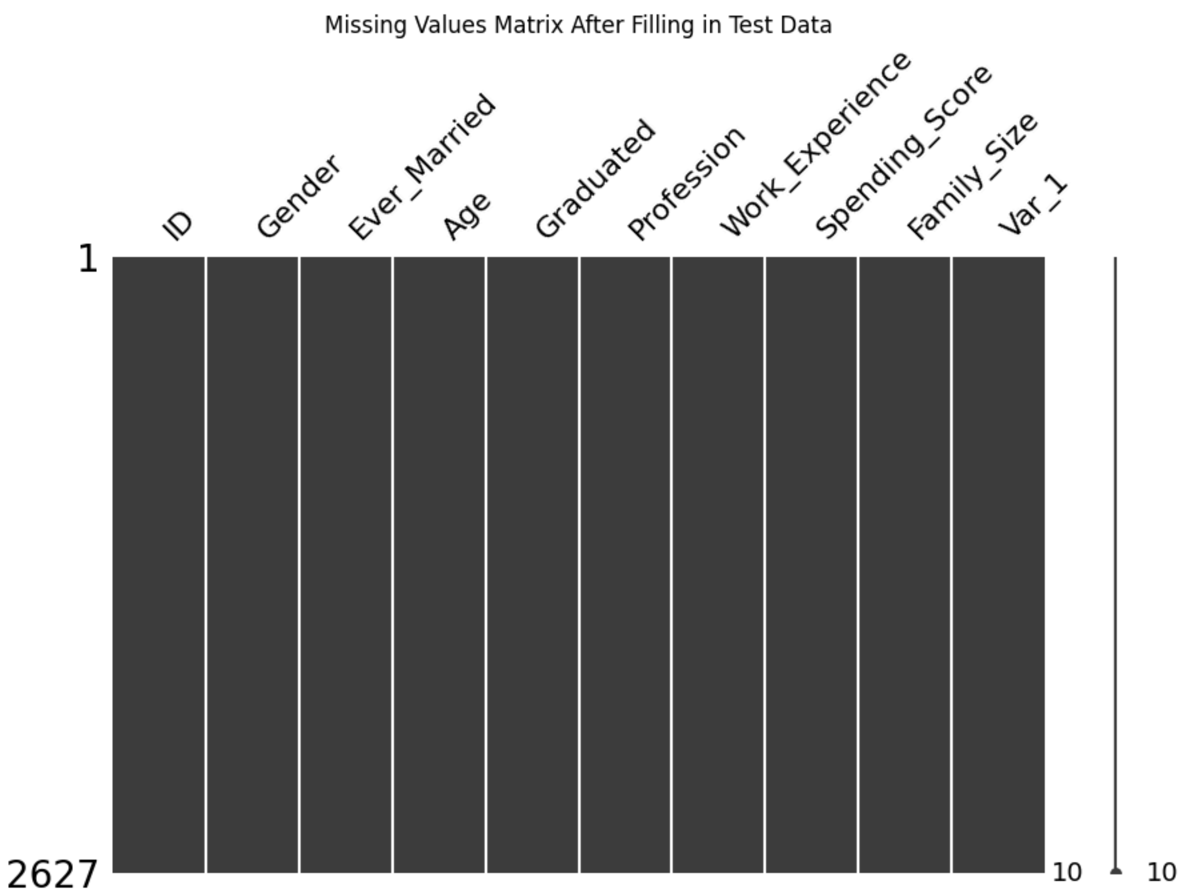





2627
7
10

```
# Copy the original datasets to new variables
train_df_copy = train_df.copy()
test_df_copy = test_df.copy()

# Handle missing values for categorical variables (mode)
for df in [train_df_copy, test_df_copy]:
    categorical_columns = df.select_dtypes(include=['object']).columns
    for column in categorical_columns:
        df[column].fillna(df[column].mode()[0], inplace=True)

# Handle missing values for numerical variables (median)
for df in [train_df_copy, test_df_copy]:
    numerical_columns = df.select_dtypes(include=['float64', 'int64']).columns
    for column in numerical_columns:
        df[column].fillna(df[column].median(), inplace=True)

# Visualize missing values for both train and test datasets
# Train dataset visualization
msno.matrix(train_df_copy, figsize=(10, 6))
plt.title('Missing Values Matrix After Filling in Train Data')
plt.show()

# Test dataset visualization
msno.matrix(test_df_copy, figsize=(10, 6))
plt.title('Missing Values Matrix After Filling in Test Data')
plt.show()

```





Perform encoding on categorical variables and scaling on numerical features and Prepare the Data

For train.csv:

▪ Split the data into X_train (features) and y_train (target labels).

For test.csv:

▪ Prepare the features (X_test), ensuring that all preprocessing steps (scaling, encoding) applied to train.csv are replicated on test.csv.

```
categorical_columns = X_train.select_dtypes(include=['object']).columns  # Categorical columns
numerical_columns = X_train.select_dtypes(include=['float64', 'int64']).columns  # Numerical columns

#Create a preprocessor that applies OneHotEncoder to categorical columns and StandardScaler to numerical columns
preprocessor = ColumnTransformer(
    transformers=[
        ('num', Pipeline(steps=[
            ('imputer', SimpleImputer(strategy='median')),  # Impute missing numerical data with median
            ('scaler', StandardScaler())  # Scale numerical data
        ]), numerical_columns),

        ('cat', Pipeline(steps=[
            ('imputer', SimpleImputer(strategy='most_frequent')),  # Impute missing categorical data with most frequent
            ('encoder', OneHotEncoder(drop='first'))  # OneHotEncode categorical data
        ]), categorical_columns)
    ])

#Apply the preprocessor to X_train and fit the transformer
X_train_prepared = preprocessor.fit_transform(X_train)

#Apply the same preprocessor to the test data (X_test)
X_test_prepared = preprocessor.transform(test_data)  # Only transform, do not fit on test data

#Check the shapes of the prepared data
print(f'Shape of X_train_prepared: {X_train_prepared.shape}')
print(f'Shape of X_test_prepared: {X_test_prepared.shape}')
```

```
Shape of X_train_prepared: (8068, 23)
Shape of X_test_prepared: (2627, 23)
```

```
# One-hot encode the target labels
y_train_transformed = pd.get_dummies(y_train).values

# Display the shape of the encoded labels
print("Shape of y_train_transformed:", y_train_transformed.shape)
```

```
Shape of y_train_transformed: (8068, 4)
```

Model 1 - Best Practices Model

Build a neural network model with at least two hidden layers using the ReLU activation function.

Use softmax as the activation function for the output layer to handle multi-class classification.

Use categorical cross-entropy as the loss function and Adam optimizer.

```
model = Sequential([
    Dense(64, activation='relu', input_shape=(X_train_prepared.shape[1],)),  # First hidden layer
    Dense(32, activation='relu'),  # Second hidden layer
    Dense(y_train_transformed.shape[1], activation='softmax')  # Output layer
])

# Step 4: Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/layers/core/dense.py:87: UserWarning:

Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the
```

Model Training:

Train the model using the train.csv data (X_train, y_train) for 50 epochs with a batch size of 32.

Use early stopping to prevent overfitting (monitor the validation loss and set a patience of 5 epochs).

Use 20% of the training data as a validation set during training

```
# Import necessary libraries
from tensorflow.keras.callbacks import EarlyStopping

# Step 1: Define early stopping callback
early_stopping = EarlyStopping(
    monitor='val_loss',
    patience=5,
    restore_best_weights=True
)

# Step 2: Train the model with early stopping
history = model.fit(
    X_train_prepared,
    y_train_transformed,
    epochs=50,
    batch_size=32,
    validation_split=0.2,
    callbacks=[early_stopping],
    verbose=1
)


```

```
Epoch 1/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 6s 5ms/step - accuracy: 0.3914 - loss: 1.2837 - val_accuracy: 0.5031 - val_loss: 1.1336
Epoch 2/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - accuracy: 0.5061 - loss: 1.1118 - val_accuracy: 0.4957 - val_loss: 1.0961
Epoch 3/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.5248 - loss: 1.0896 - val_accuracy: 0.5161 - val_loss: 1.0869
Epoch 4/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - accuracy: 0.5352 - loss: 1.0566 - val_accuracy: 0.5130 - val_loss: 1.0804
Epoch 5/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - accuracy: 0.5251 - loss: 1.0611 - val_accuracy: 0.5130 - val_loss: 1.0786
Epoch 6/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - accuracy: 0.5432 - loss: 1.0429 - val_accuracy: 0.5124 - val_loss: 1.0747
Epoch 7/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 2s 5ms/step - accuracy: 0.5439 - loss: 1.0387 - val_accuracy: 0.5124 - val_loss: 1.0698
Epoch 8/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.5515 - loss: 1.0198 - val_accuracy: 0.5217 - val_loss: 1.0721
Epoch 9/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - accuracy: 0.5580 - loss: 1.0187 - val_accuracy: 0.5198 - val_loss: 1.0715
Epoch 10/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 3s 9ms/step - accuracy: 0.5566 - loss: 1.0214 - val_accuracy: 0.5186 - val_loss: 1.0662
Epoch 11/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.5519 - loss: 1.0119 - val_accuracy: 0.5118 - val_loss: 1.0661
Epoch 12/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.5685 - loss: 1.0028 - val_accuracy: 0.5211 - val_loss: 1.0674
Epoch 13/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - accuracy: 0.5617 - loss: 1.0173 - val_accuracy: 0.5242 - val_loss: 1.0690
Epoch 14/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - accuracy: 0.5587 - loss: 1.0040 - val_accuracy: 0.5198 - val_loss: 1.0640
Epoch 15/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - accuracy: 0.5667 - loss: 0.9941 - val_accuracy: 0.5266 - val_loss: 1.0709
Epoch 16/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.5774 - loss: 0.9750 - val_accuracy: 0.5310 - val_loss: 1.0711
Epoch 17/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 2s 6ms/step - accuracy: 0.5706 - loss: 0.9950 - val_accuracy: 0.5242 - val_loss: 1.0798
Epoch 18/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.5674 - loss: 0.9838 - val_accuracy: 0.5149 - val_loss: 1.0694
Epoch 19/50
202/202 ━━━━━━━━━━━━━━━━━━━━ 2s 7ms/step - accuracy: 0.5797 - loss: 0.9561 - val_accuracy: 0.5149 - val_loss: 1.0688
```

Model Summary:

Print the model architecture summary to review the structure.

```
# Print the model architecture summary
model.summary()
# Show the model structure
plot_model(model, to_file='model_1_structure.png', show_shapes=True, show_layer_names=True)
from IPython.display import Image
Image(filename='model_1_structure.png')
```