Matplotlib Visualization Code

```
import matplotlib.pyplot as plt
import numpy as np

# Define your data
age = [47, 36, 43, 25, 52, 23, 38, 22, 57, 36,
       32, 51, 21, 26, 21, 20, 51, 31, 60, 53,
       42, 58, 33, 64, 53, 38, 31, 43, 38, 45,
       43, 55, 41, 51, 48, 42, 45, 41, 50, 61,
       43, 58, 47, 37, 53, 48, 35, 44, 37, 40,
       32, 25, 32, 52, 19, 34, 60, 44, 52, 24,
       48, 44, 66, 64, 20, 42, 32, 39, 41, 34,
       45, 29, 27, 30, 28, 31, 33]

iq = [81, 104, 108, 106, 102, 104, 100, 98, 112, 71,
      102, 93, 87, 100, 100, 110, 110, 101, 102, 108,
      108, 98, 104, 114, 106, 87, 106, 109, 108, 85,
      104, 102, 112, 100, 106, 100, 95, 89, 108, 83,
      106, 104, 114, 98, 93, 93, 100, 110, 104, 85,
      102, 102, 106, 106, 98, 116, 97, 104, 89, 110,
      102, 93, 98, 97, 105, 110, 95, 99, 96, 103,
      100, 104, 101, 105, 99, 107, None]  # Note the None value here

group = ['HC', 'AVH-', 'AVH+', 'HC', 'AVH-', 'AVH+',
         'AVH-', 'HC', 'AVH+', 'HC', 'AVH-', 'AVH-',
         'HC', 'AVH+', 'AVH-', 'AVH-', 'AVH+', 'HC',
         'AVH-', 'AVH-', 'HC', 'AVH-', 'AVH+', 'HC',
         'AVH-', 'AVH+', 'HC', 'AVH-', 'HC', 'AVH+',
         'AVH-', 'AVH+', 'HC', 'AVH-', 'AVH+', 'AVH+',
         'HC', 'AVH-', 'HC', 'AVH+', 'HC', 'AVH-',
         'AVH+', 'AVH+', 'HC', 'AVH-', 'HC', 'AVH+',
         'AVH+', 'AVH+', 'HC', 'HC', 'AVH+', 'AVH+',
         'AVH-', 'HC', 'AVH-', 'HC', 'AVH+', 'AVH-',
         'HC', 'AVH+', 'AVH+', 'AVH+', 'AVH-', 'AVH-',
         'AVH-', 'HC', 'AVH+', 'AVH-', 'HC', 'HC', 'AVH-', 'AVH+', 'HC', 'AVH-', 'AVH+']

gender = ['male', 'female', 'female', 'male', 'female', 'female',
          'male', 'male', 'female', 'male', 'female', 'female',
          'male', 'female', 'female', 'female', 'male', 'female',
          'female', 'male', 'male', 'female', 'female', 'male',
          'male', 'female', 'male', 'female', 'male', 'female',
          'female', 'female', 'male', 'female', 'female', 'female',
          'male', 'male', 'male', 'female', 'male', 'female',
```

```
        'female', 'female', 'male', 'female', 'male', 'female',
        'female', 'female', 'male', 'female', 'female', 'male',
        'female', 'female', 'male', 'male', 'male', 'male',
        'female', 'female', 'female', 'male', 'female', 'female',
        'male', 'female', 'male', 'female', 'female', 'female',
        'male', 'male', 'male', 'male', 'female']

# Remove None values from iq, along with corresponding values from age and group
filtered_data = [(a, i, g) for a, i, g in zip(age, iq, group) if i is not None]
age, iq, group = zip(*filtered_data)

# A. Distribution of Age
plt.figure(figsize=(12, 5))
plt.hist(age, bins=15, alpha=0.7, color='blue', edgecolor='black')
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()

# B. Box Plot for IQ
plt.figure(figsize=(12, 5))
plt.boxplot(iq, vert=False)
plt.title('Box Plot of IQ Scores')
plt.xlabel('IQ')
plt.grid(True)
plt.show()

# C. Gender Distribution
gender_counts = {g: gender.count(g) for g in set(gender)}
plt.figure(figsize=(8, 5))
plt.bar(gender_counts.keys(), gender_counts.values(), color=['blue', 'pink'])
plt.title('Gender Distribution')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.grid(True)
plt.show()

# D. Group Distribution
group_counts = {g: group.count(g) for g in set(group)}
plt.figure(figsize=(8, 5))
plt.bar(group_counts.keys(), group_counts.values(), color=['orange', 'green', 'red'])
plt.title('Group Distribution')
plt.xlabel('Group')
```

```
plt.ylabel('Count')
plt.grid(True)
plt.show()

# E. Age vs IQ Scatter Plot
plt.figure(figsize=(12, 5))
scatter = plt.scatter(age, iq, c=[{'HC': 0, 'AVH-': 1, 'AVH+': 2}[g] for g in group], cmap='viridis',
edgecolor='k')
plt.title('Age vs IQ by Group')
plt.xlabel('Age')
plt.ylabel('IQ')
plt.colorbar(scatter, ticks=[0, 1, 2], label='Group')
plt.grid(True)
plt.show()

# F. Confusion Matrix for Predictions (Define y_test_balanced and y_pred_balanced before this)
# Assuming y_test_balanced and y_pred_balanced are defined, you can create a confusion
matrix
conf_matrix = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]
# Example initialization (use real predictions)
y_test_balanced = ['HC', 'AVH-', 'AVH+', 'HC']  # Example true labels
y_pred_balanced = ['HC', 'AVH-', 'HC', 'AVH+']  # Example predictions
for true, pred in zip(y_test_balanced, y_pred_balanced):
    true_index = ['HC', 'AVH-', 'AVH+'].index(true)
    pred_index = ['HC', 'AVH-', 'AVH+'].index(pred)
    conf_matrix[true_index][pred_index] += 1

plt.figure(figsize=(10, 7))
plt.imshow(conf_matrix, interpolation='nearest', cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted Group')
plt.ylabel('True Group')
plt.xticks(np.arange(3), ['HC', 'AVH-', 'AVH+'])
plt.yticks(np.arange(3), ['HC', 'AVH-', 'AVH+'])
plt.colorbar()
for i in range(3):
    for j in range(3):
        plt.text(j, i, conf_matrix[i][j], ha='center', va='center', color='black')
plt.grid(False)
plt.show()

# G. Performance by Gender (Define gender_perf_after before this)
# Assuming gender_perf_after is defined
gender_perf_after = {'male': 0.85, 'female': 0.75}  # Example performance data
```

```
plt.figure(figsize=(8, 5))
plt.bar(gender_perf_after.keys(), gender_perf_after.values(), color=['blue', 'pink'])
plt.title('Model Performance by Gender')
plt.xlabel('Gender')
plt.ylabel('Accuracy')
plt.grid(True)
plt.show()
```