1 is left unilateral

2 is right unilateral

3 is bilateral

In [ ]:
```
import pandas as pd

df = pd.read_csv('updated_complete_vol_visual.csv')
```

In [ ]:
```
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score, KFold
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.preprocessing import StandardScaler

PCA = df
PCA = PCA[PCA['PCA'] != 0]

X = pd.concat([PCA['PCAL_x'], PCA['PCAR_x']], axis=1)
y = PCA['PCA']

# Set up cross-validation:
kf = KFold(n_splits=5, shuffle=True, random_state=42)

model = LogisticRegression(multi_class='multinomial', solver='lbfgs', max_it

# Perform cross-validation:
cv_scores = cross_val_score(model, X, y, cv=kf)

# Print cross-validated scores:
print("Cross-validated scores:", cv_scores)
print("Average cross-validation score:", cv_scores.mean())

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, rar
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

# Evaluate the model:
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

print("Accuracy of the model on the test set:", accuracy)
print("")
print("Confusion Matrix:\n", conf_matrix)

# Create a DataFrame with the test set and predictions:
results_df = pd.DataFrame(X_test)
results_df['Actual'] = y_test
results_df['Predicted'] = y_pred

# Filter the DataFrame to only include misclassified instances:
misclassified = results_df[results_df['Actual'] != results_df['Predicted']]

print("")
print("Misclassified instances:")
print(misclassified)
```

```
Cross-validated scores: [0.92857143 0.92857143 0.92857143 0.88888889 1.
]
Average cross-validation score: 0.9349206349206348
Accuracy of the model on the test set: 0.9285714285714286

Confusion Matrix:
 [[12  0  0]
 [ 0 13  0]
 [ 1  1  1]]

Misclassified instances:
     PCAL_x  PCAR_x  Actual  Predicted
410     975   10145     3.0        2.0
405     959     112     3.0        1.0
```

In [ ]:
```
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score, KFold
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.preprocessing import StandardScaler

Thalamus = df
Thalamus = Thalamus[Thalamus['PCA'] != 0]

X = pd.concat([Thalamus['ThalamusL_x'], Thalamus['ThalamusR_x']], axis=1)
y = Thalamus['Thalamus']

# Set up cross-validation:
kf = KFold(n_splits=5, shuffle=True, random_state=42)

model = LogisticRegression(multi_class='multinomial', solver='lbfgs', max_it

# Perform cross-validation:
cv_scores = cross_val_score(model, X, y, cv=kf)

# Print cross-validated scores:
print("Cross-validated scores:", cv_scores)
print("Average cross-validation score:", cv_scores.mean())

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, rar
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

# Evaluate the model:
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

print("Accuracy of the model on the test set:", accuracy)
print("")
print("Confusion Matrix:\n", conf_matrix)

# Create a DataFrame with the test set and predictions:
results_df = pd.DataFrame(X_test)
results_df['Actual'] = y_test
results_df['Predicted'] = y_pred

# Filter the DataFrame to only include misclassified instances:
misclassified = results_df[results_df['Actual'] != results_df['Predicted']]

print("")
print("Misclassified instances:")
print(misclassified)
```

```
Cross-validated scores: [0.82142857 0.75       0.82142857 0.92592593 0.9259
2593]
Average cross-validation score: 0.8489417989417989
Accuracy of the model on the test set: 0.8214285714285714

Confusion Matrix:
 [[18  0  2]
 [ 0  3  0]
 [ 3  0  2]]

Misclassified instances:
     ThalamusL_x  ThalamusR_x  Actual  Predicted
406            0          161     2.0        0.0
87             0          473     2.0        0.0
411            0          343     2.0        0.0
24             0          543     0.0        2.0
414            0          876     0.0        2.0
```

In [ ]:
```
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score, KFold
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.preprocessing import StandardScaler

cerebellum = df
cerebellum = cerebellum[cerebellum['PCA'] != 0]

X = pd.concat([cerebellum['cerebellumL_x'], cerebellum['cerebellumR_x']], ax
y = cerebellum['cerebellum']

# Set up cross-validation:
kf = KFold(n_splits=5, shuffle=True, random_state=42)

model = LogisticRegression(multi_class='multinomial', solver='lbfgs', max_it

# Perform cross-validation:
cv_scores = cross_val_score(model, X, y, cv=kf)

# Print cross-validated scores:
print("Cross-validated scores:", cv_scores)
print("Average cross-validation score:", cv_scores.mean())

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ran
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

# Evaluate the model:
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

print("Accuracy of the model on the test set:", accuracy)
print("")
print("Confusion Matrix:\n", conf_matrix)

# Create a DataFrame with the test set and predictions:
results_df = pd.DataFrame(X_test)
results_df['Actual'] = y_test
results_df['Predicted'] = y_pred

# Filter the DataFrame to only include misclassified instances:
misclassified = results_df[results_df['Actual'] != results_df['Predicted']]

print("")
print("Misclassified instances:")
print(misclassified)
```

```
Cross-validated scores: [0.92857143 0.96428571 1.         0.85185185 0.8888
8889]
Average cross-validation score: 0.9267195767195766
Accuracy of the model on the test set: 0.9285714285714286

Confusion Matrix:
 [[24  0  0]
 [ 0  0  1]
 [ 1  0  2]]

Misclassified instances:
     cerebellumL_x  cerebellumR_x  Actual  Predicted
406          31895            461     2.0        3.0
405            578              0     3.0        0.0
```