Explainable EEG Network Analysis for Alzheimer's Disease Detection: Node-Level Graph Features and XGBoost

Hongtao Ma

2026-02-03

1. Project Overview

This project analyzes a publicly available EEG dataset related to Alzheimer's disease (AD), released through the **UCI Machine Learning** Repository and hosted on Kaggle as the **"EEG Alzheimer's Dataset"**:

Data Address: https://www.kaggle.com/datasets/ucimachinelearning/eeg-alzheimers-dataset

Electroencephalography (EEG) is a non-invasive neurophysiological technique that measures the brain's electrical activity through electrodes placed on the scalp. EEG signals provide a high-temporal-resolution view of neural dynamics and are widely used in clinical and cognitive neuroscience research, including the study of neurodegenerative disorders such as Alzheimer's disease.

The dataset contains **848,640 EEG samples** with **17 columns**, including **16 continuous EEG channel features** and one categorical target label. The 16 channels correspond to standard electrode placements under the international **10–20 EEG system**, covering multiple cortical regions:

- **Frontal region:** Fp1, Fp2, F7, F3, Fz, F4, F8
- **Central region:** T3, C3, Cz, C4, T4
- **Parietal/Temporal-posterior region:** T5, P3, Pz, P4

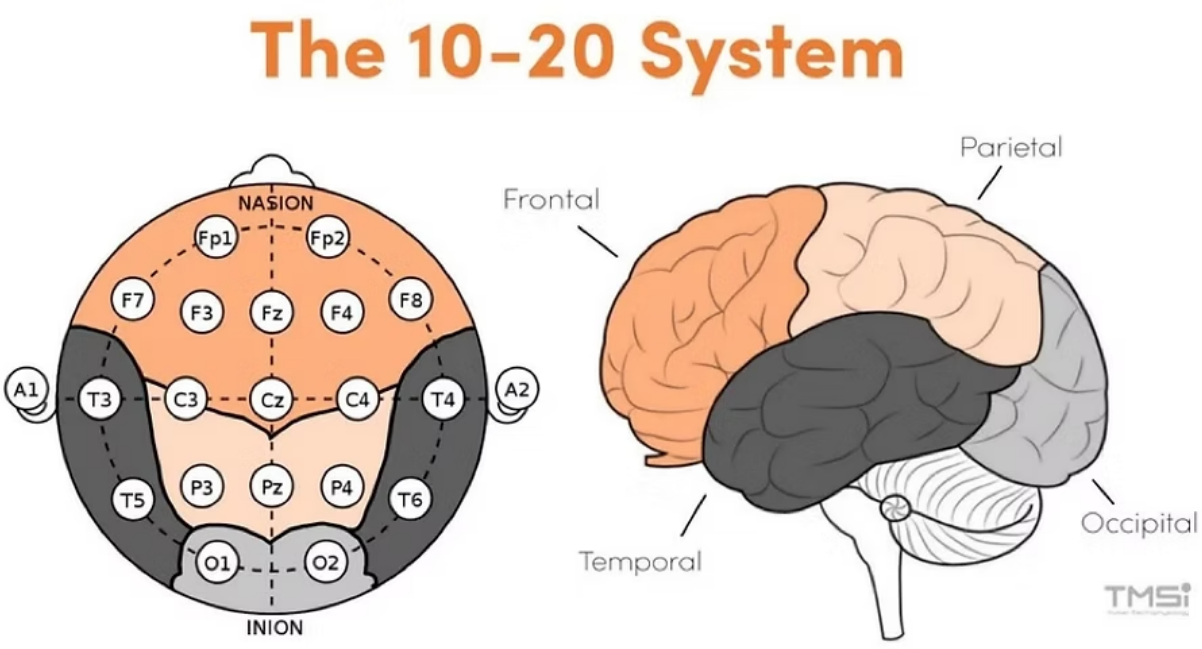

Figure 01: Diagram 10–20 EEG system

Source: https://www.integrisneuro.com/post/10-20-measurement-system

---

These channels reflect brain activity across different lobes (frontal, temporal, central, and parietal), enabling investigation of region-specific network alterations associated with AD.

The variable, **status**, indicates whether the EEG recording corresponds to an Alzheimer's disease patient or a healthy control subject. The dataset is clean, with no missing values, making it suitable for machine learning-based EEG classification and network feature extraction.

Objective. The primary goal of this project is to develop an accurate and interpretable model for distinguishing **AD vs. healthy controls (HC)** using **network-based EEG analysis**. Specifically, the workflow converts windowed EEG signals into functional connectivity graphs and extracts node-level graph metrics for each channel. This allows the final results to identify not only overall classification performance, but also **which EEG regions contribute most strongly to AD-related network disruptions** and how their connectivity roles differ between AD and HC.

---

2. Data preparation & Experimental Design

First, let us import the data:

```
library(data.table)
library(igraph)
library(caret)
library(xgboost)
library(pROC)

data_path <- "D:/harvard_edx/EEG alzheimer/AD_all_patients.csv"
data <- fread(data_path)

eeg_cols <- c(
  "Fp1","Fp2","F7","F3","Fz","F4","F8",
  "T3","C3","Cz","C4","T4",
  "T5","P3","Pz","P4"
)

X_raw <- data[, ..eeg_cols]
y_raw <- data$status  # 0=HC, 1=AD

table(y_raw)
```

```
## y_raw
##      0      1
## 336640 512000
```

```
summary(X_raw)
```

```
##       Fp1                Fp2                F7                F3
##  Min.   :-698.175   Min.   :-555.575   Min.   :-334.40   Min.   :-460.775
##  1st Qu.: -25.956   1st Qu.: -10.381   1st Qu.: -38.18   1st Qu.: -19.387
##  Median : -11.800   Median :   2.256   Median :  -7.10   Median :  -1.056
##  Mean   :   9.198   Mean   :  23.204   Mean   : -11.43   Mean   :   4.301
##  3rd Qu.:  23.269   3rd Qu.:  35.487   3rd Qu.:  20.87   3rd Qu.:  19.387
##  Max.   :7077.325   Max.   : 376.938   Max.   : 409.14   Max.   : 638.925
##        Fz                 F4                 F8                 T3
##  Min.   :-434.975   Min.   :-572.075   Min.   :-1000.331   Min.   :-1006.575
##  1st Qu.: -42.244   1st Qu.: -39.475   1st Qu.:  -22.769   1st Qu.:  -55.869
##  Median : -13.931   Median : -18.025   Median :    2.763   Median :   -8.694
##  Mean   : -26.752   Mean   : -19.904   Mean   :  -11.155   Mean   :  -21.951
##  3rd Qu.:   5.645   3rd Qu.:   7.362   3rd Qu.:   29.075   3rd Qu.:   13.744
##  Max.   : 533.962   Max.   : 227.188   Max.   : 1152.419   Max.   :  281.562
##        C3                 Cz                 C4                 T4
##  Min.   :-465.375   Min.   :-163.744   Min.   :-451.425   Min.   :-451.675
##  1st Qu.:  -2.606   1st Qu.: -11.088   1st Qu.:  -9.300   1st Qu.:  -4.919
##  Median :  12.919   Median :   1.766   Median :   7.188   Median :   6.781
##  Mean   :  18.052   Mean   :  18.132   Mean   :  16.904   Mean   :  17.302
##  3rd Qu.:  29.700   3rd Qu.:  24.175   3rd Qu.:  39.981   3rd Qu.:  23.875
##  Max.   : 346.969   Max.   : 805.356   Max.   : 343.512   Max.   : 358.931
##        T5                 P3                 Pz                 P4
##  Min.   :-726.37   Min.   :-487.075   Min.   :-385.775   Min.   :-646.413
##  1st Qu.: -29.02   1st Qu.: -21.413   1st Qu.:   4.906   1st Qu.: -36.906
##  Median :  -3.40   Median :  -6.225   Median :  24.062   Median : -12.569
##  Mean   : -13.82   Mean   :  -4.301   Mean   :  39.720   Mean   : -37.499
##  3rd Qu.:  11.91   3rd Qu.:  11.425   3rd Qu.:  72.312   3rd Qu.:   4.425
##  Max.   : 244.69   Max.   : 214.700   Max.   : 595.250   Max.   : 115.894
```

2.1 Anti-leakage Grouping (Conservative)

Neighboring EEG samples are correlated and often belong to the same underlying continuous recording segment.

To reduce leakage between train and test windows, we create a conservative `group_id` that increments whenever the label changes.

> **Note:** If the raw dataset contains subject IDs, subject-based grouping would be preferred and more principled.

```
candidate_id_cols <- c("subject_id","SubjectID","patient_id","PatientID","sub_id","id","ID")
id_col <- candidate_id_cols[candidate_id_cols %in% names(data)][1]

if (!is.na(id_col)) {
  group_id <- as.integer(as.factor(data[[id_col]]))
  cat("Using subject/patient id column as group:", id_col, "\n")
} else {
  group_id <- rleid(y_raw)   # data.table::rleid
  cat("WARNING: No subject/patient id column found; using rleid(status) as group.\n")
}
```

```
## WARNING: No subject/patient id column found; using rleid(status) as group.
```

```
length(unique(group_id))
```

```
## [1] 18
```

2.2 Window-based segmentation

At the same time, EEG signals are **high-dimensional, temporally correlated, and highly non-stationary**, making direct application of machine learning to raw time series often **unstable and difficult to interpret**.

To address this, we adopt a **network-based representation** of EEG data. The continuous signals are segmented into **short, overlapping time windows**, which allows us to capture **dynamic functional connectivity patterns** while maintaining **reliable correlation estimates**.

We use a standard sliding window approach:

- Window length = 512 samples
- Step size = 256 samples (50% overlap)

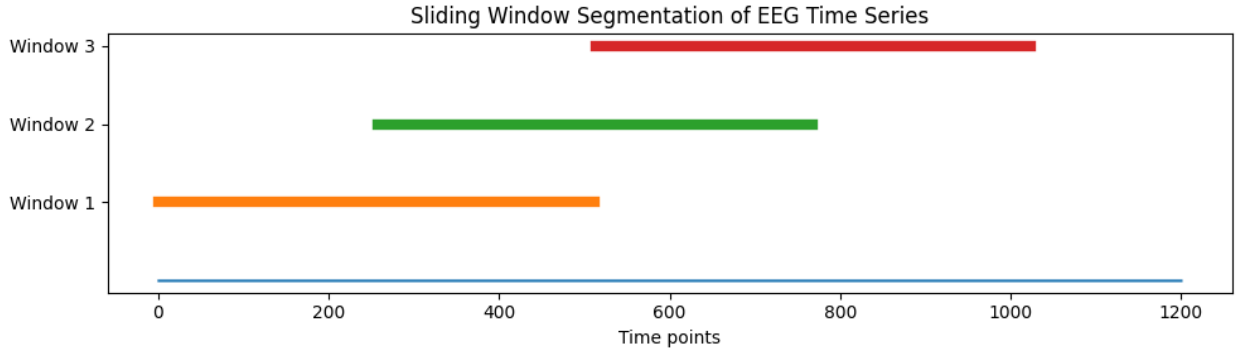

Figure 02: Diagram of Window Segmentation

Each window is treated as one sample for subsequent network construction and modeling.

Reference: https://en.wikipedia.org/wiki/Dynamic_functional_connectivity

```
window_length <- 512
step_size <- 256

windows <- list()

for (g in unique(group_id)) {
  idx <- which(group_id == g)
  if (length(idx) < window_length) next

  starts <- seq(1, length(idx) - window_length + 1, by = step_size)
  for (s in starts) {
    sel <- idx[s:(s + window_length - 1)]
```

```
    windows[[length(windows) + 1]] <- list(
      signal = as.matrix(X_raw[sel]),
      label  = y_raw[sel[1]],
      group  = g
    )
  }
}

cat("Number of windows:", length(windows), "\n")
```

```
## Number of windows: 3297
```

2.3 Functional brain network construction

After window segmentation, we compute a **functional connectivity matrix** across the 16 EEG channels for each window using **Pearson correlation**.

Let x_i and x_j denote the EEG time series of channels i and j within a given window. The functional connectivity between channels is defined as:

$$C_{ij} = \text{corr}(x_i, x_j)$$

This results in a 16×16 symmetric correlation matrix $\mathbf{C}$, where each entry reflects the linear statistical dependency between two EEG channels within the same temporal window.

```
compute_connectivity <- function(window_signal) {
  cor(window_signal, method = "pearson", use = "pairwise.complete.obs")
}
```

2.4 Node-level graph feature extraction (channel-specific biomarkers)

To enable **region-specific interpretation** of Alzheimer's disease–related network alterations, we extract **node-level graph-theoretic features** from each **functional connectivity matrix** rather than relying solely on global network metrics.

For each temporal window, the **functional connectivity matrix C** is converted into a **weighted undirected brain network**, where:

- **Nodes** correspond to individual EEG channels
- **Edge weights** are defined as the absolute correlation magnitude:

$$W_{ij} = |C_{ij}|$$

This formulation captures the **strength of functional coupling** between brain regions while remaining **agnostic to correlation sign**.

For **distance-based graph measures** (e.g., closeness and betweenness centrality), edge weights are transformed into distances as:

$$d_{ij} = \frac{1}{W_{ij}}$$

Using this weighted graph representation, we compute the following **node-level topological features** for each EEG channel:

- **Strength** – overall connectivity intensity of a node
- **Clustering coefficient** – degree of local neighborhood clustering
- **Closeness centrality** – global reachability and integration efficiency
- **Betweenness centrality** – importance of a node as a communication bridge
- **Local efficiency** – efficiency of information transfer within a node's immediate neighborhood
- **Participation coefficient** – extent of cross-community connectivity

Together, these features characterize how individual brain regions participate in large-scale functional networks and enable systematic comparison of **network roles between Alzheimer's disease (AD) patients and healthy control (HC) subjects**.

```
make_graph_from_C <- function(C, eps = 1e-6) {
  diag(C) <- 0
  W <- abs(C)
  W[W < eps] <- eps
  g <- graph_from_adjacency_matrix(W, mode = "undirected", weighted = TRUE, diag = FALSE)
  E(g)$dist <- 1 / E(g)$weight
  g
}

global_eff_from_dist <- function(g) {
  D <- distances(g, weights = E(g)$dist)
  diag(D) <- NA
  mean(1 / D, na.rm = TRUE)
}

local_efficiency_node <- function(g, vname) {
  v <- which(V(g)$name == vname)
  nb <- neighbors(g, v)
  if (length(nb) < 2) return(NA_real_)
  sg <- induced_subgraph(g, nb)
  global_eff_from_dist(sg)
}

extract_node_features <- function(C, channels) {
  g <- make_graph_from_C(C)
  V(g)$name <- channels

  str   <- strength(g, weights = E(g)$weight)
  clust <- transitivity(g, type = "local", isolates = "zero")
  close <- closeness(g, weights = E(g)$dist, normalized = TRUE)
  btw   <- betweenness(g, weights = E(g)$dist, normalized = TRUE)
  leff  <- sapply(channels, function(ch) local_efficiency_node(g, ch))

  comm <- cluster_fast_greedy(g, weights = E(g)$weight)
  memb <- membership(comm)
```

```r
  part <- sapply(channels, function(ch) {
    v <- which(V(g)$name == ch)
    nb <- neighbors(g, v)
    if (length(nb) == 0) return(0)

    ki <- str[ch]
    if (is.na(ki) || ki == 0) return(0)

    eids <- incident(g, v, mode = "all")
    ends_mat <- ends(g, eids)
    nb_names <- V(g)$name[as.integer(nb)]

    keep <- (ends_mat[,1] == ch & ends_mat[,2] %in% nb_names) |
      (ends_mat[,2] == ch & ends_mat[,1] %in% nb_names)

    w <- E(g)[eids[keep]]$weight
    nb2 <- ifelse(ends_mat[keep,1] == ch, ends_mat[keep,2], ends_mat[keep,1])
    nb_comm <- memb[match(nb2, V(g)$name)]

    w_to_c <- tapply(w, nb_comm, sum)
    1 - sum((w_to_c / ki)^2, na.rm = TRUE)
  })

  feats <- c()
  for (ch in channels) {
    feats[paste0(ch, "_strength")]      <- str[ch]
    feats[paste0(ch, "_clust")]         <- clust[ch]
    feats[paste0(ch, "_closeness")]     <- close[ch]
    feats[paste0(ch, "_betweenness")]   <- btw[ch]
    feats[paste0(ch, "_local_eff")]     <- leff[ch]
    feats[paste0(ch, "_participation")] <- part[ch]
  }
  feats
}
```

2.5 Feature Table Construction (Window-level)

Through the above procedure, the **original EEG time-series data**, as illustrated in the following figure,

time	Fp1	Fp2	F7	...	P4
t1	116	41	66	...	-245
t2	115	38	64	...	-248
...	...	...	...	...	...
t512	120	45	74	...	-236

Figure 03: Example of a single EEG time window (512 time points × 16 channels)

are transformed into a **structured sample-level dataset** consisting of **16 × 6 = 96 network features**, together with **2 additional columns** for the class label and group identifier.

Sample	Fp1_strength	Fp1_clust	...	P4_participation	label	group
1	3.92	0.41	...	0.67	AD	12
2	3.85	0.38	...	0.70	AD	12
3	2.91	0.55	...	0.42	HC	13
...	...	...	...	...	...	...

Figure 04: Example structure of the feature table

```
feature_list <- lapply(windows, function(w) {
  C <- compute_connectivity(w$signal)
  node_feats <- extract_node_features(C, eeg_cols)
  c(node_feats, label = w$label, group = w$group)
})

feature_df <- as.data.frame(do.call(rbind, feature_list))
feature_df$label <- as.factor(feature_df$label)

dim(feature_df)
```

```
## [1] 3297   98
```

```
head(feature_df[, c(1:6, (ncol(feature_df)-1):ncol(feature_df))])
```

```
##   Fp1_strength Fp1_clust Fp1_closeness Fp1_betweenness Fp1_local_eff
## 1     3.809027         1     0.3096707      0.00952381     0.5038031
## 2     4.636098         1     0.3435784      0.00952381     0.4255471
## 3     4.997543         1     0.3464952      0.00952381     0.4690970
## 4     4.211094         1     0.3200127      0.00000000     0.4732793
## 5     4.179898         1     0.3054895      0.01904762     0.3802868
## 6     6.326849         1     0.3981824      0.06666667     0.4060070
##   Fp1_participation label group
## 1         0.4924502     1     1
## 2         0.4995769     1     1
## 3         0.4754612     1     1
## 4         0.4421877     1     1
## 5         0.6350956     1     1
## 6         0.4930793     1     1
```

---

3. Modeling and Evaluation

3.1 Predictive modeling with leakage-aware validation

We train an **XGBoost classifier** to distinguish AD patients from healthy controls using the extracted node-level graph features.

But before this, to reduce over-optimistic performance caused by temporal dependence, we will apply **group-aware splitting (GroupKFold)** so that correlated windows are not simultaneously present in both training and testing sets.

```r
# ========================
# 6) Modeling: group-stratified split + XGBoost (FIXED)
# ========================
set.seed(42)

make_group_stratified_folds <- function(group, label, k = 5, seed = 42) {
  set.seed(seed)
  df_g <- unique(data.frame(group = group, label = label))
  df_g$label <- as.factor(df_g$label)

  fold_id <- caret::createFolds(df_g$label, k = k, list = FALSE)
  folds <- lapply(1:k, function(i) df_g$group[fold_id == i])
  folds
}

fold_groups <- make_group_stratified_folds(
  group = feature_df$group,
  label = feature_df$label,
  k = 5,
  seed = 42
)

test_groups  <- fold_groups[[1]]
test_idx  <- which(feature_df$group %in% test_groups)
train_idx <- which(!feature_df$group %in% test_groups)

if (length(unique(feature_df$label[test_idx])) < 2) {
  stop("Test set contains only one class after splitting. Check group/label distribution.")
}

X_train <- as.matrix(feature_df[train_idx, !names(feature_df) %in% c("label", "group")])
y_train <- feature_df$label[train_idx]

X_test <- as.matrix(feature_df[test_idx, !names(feature_df) %in% c("label", "group")])
y_test <- feature_df$label[test_idx]


dtrain <- xgb.DMatrix(X_train, label = as.numeric(y_train) - 1)
dtest  <- xgb.DMatrix(X_test,  label = as.numeric(y_test) - 1)
```

3.2 XGBoost Training

XGBoost is employed because it effectively handles **high-dimensional graph-derived EEG features**, captures **nonlinear relationships** between network properties, and incorporates regularization to mitigate overfitting arising from temporally correlated windows. Importantly, its native support for **SHAP-style explanations** facilitates interpretable mapping from model predictions to brain-region–specific network alterations.

We train a binary classifier with standard regularization settings.

```r
params <- list(
  objective = "binary:logistic",
  eval_metric = "auc",
  max_depth = 4,
  eta = 0.05,
  subsample = 0.8,
  colsample_bytree = 0.8
)

model <- xgb.train(
  params  = params,
  data    = dtrain,
  nrounds = 300,
  verbose = 0
)
```

3.3 Performance Metrics (AUC + Probability RMSE + Confusion Matrix)

To provide a comprehensive evaluation, we report metrics that reflect both classification performance and probabilistic accuracy. AUC measures threshold-independent discrimination between AD and HC, whereas the RMSE of predicted probabilities evaluates how well predicted disease risks align with observed outcomes, serving as a calibration-oriented error measure.

- **AUC** measures discrimination.
- **RMSE of predicted probabilities** measures probability error:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{p}_i)^2}$$

```r
# ========================
# 7) Metrics + ROC (FIXED)
# ========================
pred_prob  <- predict(model, dtest)
pred_class <- ifelse(pred_prob >= 0.5, 1, 0)

y_test <- factor(y_test, levels = c("0","1"))

roc_obj <- pROC::roc(response = y_test, predictor = pred_prob, levels = c("0","1"), direction = "<")
auc_value <- as.numeric(pROC::auc(roc_obj))

y_true <- as.numeric(as.character(y_test))
rmse_prob <- sqrt(mean((y_true - pred_prob)^2))

cat("AUC =", auc_value, "\n")
```

```
## AUC = 0.8213661
```

```r
cat("RMSE (probability RMSE) =", rmse_prob, "\n\n")
```

```
## RMSE (probability RMSE) = 0.5101828
```

```r
cm <- table(Predicted = pred_class, Actual = y_true)
cm
```

```
##          Actual
## Predicted   0   1
##         0 138  20
##         1 190 203
```

```r
plot(roc_obj, main = "ROC Curve: AD vs HC (Node-level Network Features)",
     col = "blue", lwd = 2)
```

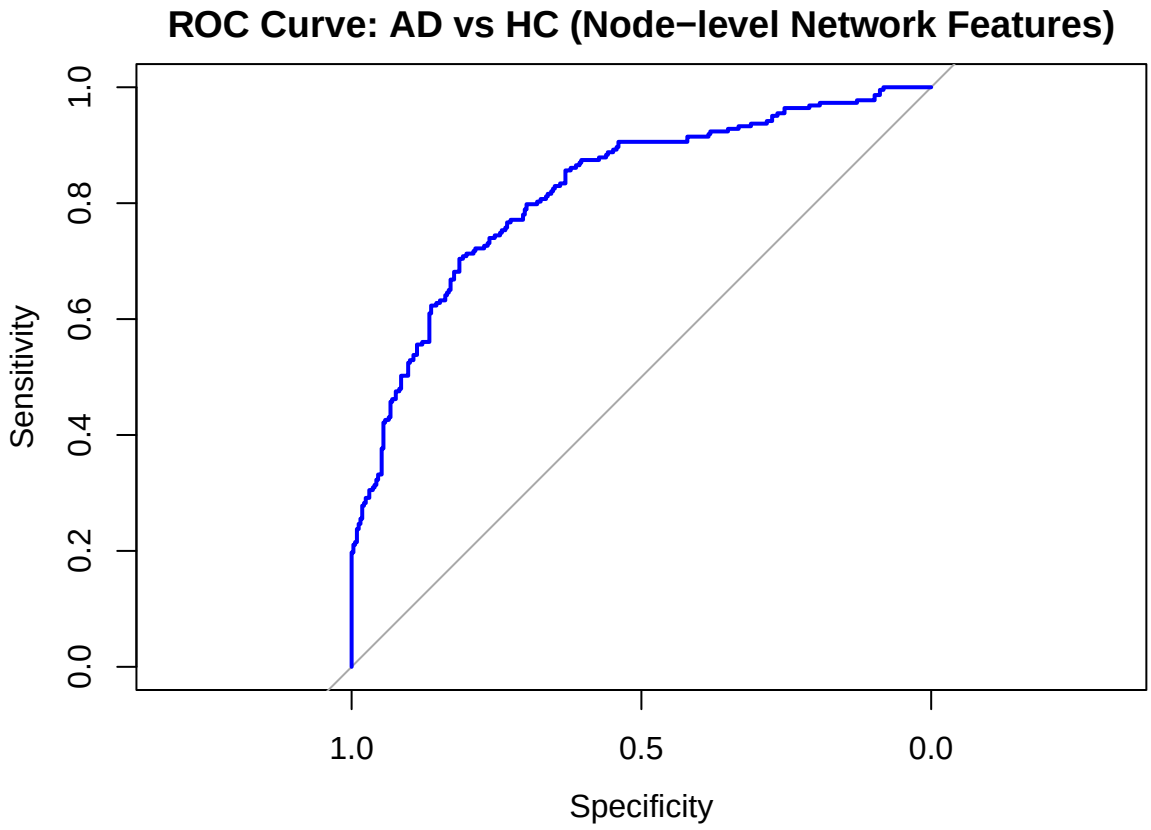

Figure 05: ROC Curve: AD vs HC (Node—level Network Features)

The Figure 05 shows the ROC curve for distinguishing **Alzheimer's disease (AD) patients from healthy controls (HC)** using node-level EEG network features. The curve lies substantially above the diagonal reference line, indicating strong discriminative performance across a wide range of decision thresholds. This result demonstrates that the proposed network-based representation captures **disease-relevant functional connectivity patterns**, enabling reliable separation of AD and HC at the window level. Importantly, the observed discrimination supports the study's objective of leveraging **interpretable node-level graph features** to identify Alzheimer's disease–related network alterations rather than relying on raw EEG signals alone.

4. Explainability: SHAP aggregation, directionality, and effect size

To interpret the trained XGBoost model and identify **which EEG channels and network properties drive Alzheimer's disease classification**, we employ a post hoc explainable machine learning framework combining **SHAP-style feature contributions**, **directional group differences**, and **effect size estimation**.

4.1 Explainable machine learning and feature attribution strategy

Model interpretability is achieved using **SHAP-style additive feature attributions** computed directly from XGBoost via:

- `predcontrib = TRUE`, which decomposes each prediction into per-feature contributions

These feature attributions are subsequently aggregated to provide two complementary interpretability views:

- **Channel-level importance rankings**, identifying which EEG regions contribute most strongly to classification
- **Metric-level importance rankings**, identifying which node-level network properties are most informative

This hierarchical aggregation enables interpretation at both the **neuroanatomical** and **network-topological** levels.

4.2 SHAP-style feature contributions (XGBoost `predcontrib`)

For each test sample, SHAP-style feature contributions are computed using the trained XGBoost model. Feature importance is defined as the **mean absolute SHAP value** across samples:

$$\text{Importance}(f) = \mathbb{E}\left(\left|\text{SHAP}_f\right|\right)$$

```
shap_test <- predict(model, dtest, predcontrib = TRUE)
shap_test <- as.data.frame(shap_test)
shap_test$BIAS <- NULL

shap_imp <- colMeans(abs(shap_test), na.rm = TRUE)
shap_imp <- sort(shap_imp, decreasing = TRUE)

head(shap_imp, 10)
```

```
##     (Intercept)  C3_betweenness     C3_strength    P4_closeness Fp2_betweenness
##       0.5630223       0.4130434       0.3333126       0.2683830       0.2470489
##  Cz_betweenness     P4_strength    T4_closeness  Pz_betweenness  F7_betweenness
##       0.2211398       0.2152805       0.2117363       0.2114505       0.1936541
```

This measure reflects the average magnitude of a feature's contribution to model predictions, independent of direction.

4.3 Aggregate Importance to Channels and Metrics

To obtain interpretable summaries, individual feature importances are aggregated along two dimensions:

- **Channel importance:** sum of feature importances within a channel
- **Metric importance:** sum of importances across channels for a metric type

```
channel_importance <- sapply(eeg_cols, function(ch) {
  feats <- grep(paste0("^", ch, "_"), names(shap_imp), value = TRUE)
  sum(shap_imp[feats], na.rm = TRUE)
})
channel_importance <- sort(channel_importance, decreasing = TRUE)

metric_names <- c("strength", "clust", "closeness", "betweenness", "local_eff", "participation")
metric_importance <- sapply(metric_names, function(mn) {
  feats <- grep(paste0("_", mn, "$"), names(shap_imp), value = TRUE)
  sum(shap_imp[feats], na.rm = TRUE)
})
metric_importance <- sort(metric_importance, decreasing = TRUE)

cat("Top channels by SHAP importance:\n")
```

```
## Top channels by SHAP importance:
```

```
print(head(channel_importance, 10))
```

```
##        C3        Cz        P4        F7        F3        P3       Fp2        F8
## 0.8926921 0.5975413 0.5915888 0.5775922 0.5077441 0.4913822 0.4811306 0.4550065
##       Fp1        T4
## 0.4436190 0.3912999
```

```
cat("\nMetric importance (overall):\n")
```

```
##
## Metric importance (overall):
```

```
print(metric_importance)
```

```
##      strength   betweenness     closeness participation     local_eff
##     2.1745538     2.0647844     1.6380931     0.8563463     0.3279791
##         clust
##     0.0000000
```

These aggregated rankings allow identification of key brain regions and dominant network mechanisms underlying AD–HC discrimination.

4.4 Directionality of effects and Cohen's *d* effect size

To complement SHAP-based importance with **group-level interpretability**, we compute the **direction and magnitude of AD–HC differences** for each feature.

For each feature f, we compute:

- **Mean difference**:
$$\Delta_f = \text{mean}_{AD}(f) - \text{mean}_{HC}(f)$$
- **Cohen's *d*** using pooled standard deviation

```
X_all <- feature_df[, !names(feature_df) %in% c("label","group")]
lbl <- feature_df$label

mean_ad <- colMeans(X_all[lbl == "1", , drop=FALSE], na.rm = TRUE)
mean_hc <- colMeans(X_all[lbl == "0", , drop=FALSE], na.rm = TRUE)
diff_ad_minus_hc <- mean_ad - mean_hc

sd_pooled <- function(x1, x0) {
  s1 <- sd(x1, na.rm=TRUE); s0 <- sd(x0, na.rm=TRUE)
  n1 <- sum(!is.na(x1));    n0 <- sum(!is.na(x0))
  if ((n1 + n0 - 2) <= 0) return(NA_real_)
  sqrt(((n1-1)*s1^2 + (n0-1)*s0^2) / (n1 + n0 - 2))
}

cohen_d <- sapply(names(X_all), function(fn) {
  x1 <- X_all[lbl=="1", fn]
  x0 <- X_all[lbl=="0", fn]
  sp <- sd_pooled(x1, x0)
  if (is.na(sp) || sp==0) return(NA_real_)
  (mean(x1, na.rm=TRUE) - mean(x0, na.rm=TRUE)) / sp
})

top_features <- names(head(shap_imp, 15))
result_table <- data.frame(
  feature = top_features,
  shap_importance = shap_imp[top_features],
  mean_diff_AD_minus_HC = diff_ad_minus_hc[top_features],
  cohen_d = cohen_d[top_features],
  row.names = NULL
)

result_table
```

```
##             feature shap_importance mean_diff_AD_minus_HC     cohen_d
## 1       (Intercept)       0.5630223                    NA          NA
## 2    C3_betweenness       0.4130434           0.013546602  0.43345057
## 3       C3_strength       0.3333126           1.341359540  0.87081152
## 4      P4_closeness       0.2683830           0.057937226  0.73628588
## 5   Fp2_betweenness       0.2470489           0.002067369  0.09039037
## 6    Cz_betweenness       0.2211398           0.006313718  0.18323750
## 7       P4_strength       0.2152805           1.105482804  0.72835543
## 8      T4_closeness       0.2117363           0.053093716  0.67310995
```

```
## 9    Pz_betweenness       0.2114505          -0.013242952 -0.40749805
## 10   F7_betweenness       0.1936541          -0.005138099 -0.14877596
## 11      F7_strength       0.1922198           0.679978354  0.41323867
## 12      F3_strength       0.1822325           0.903686265  0.55857355
## 13   F3_betweenness       0.1790925           0.001681248  0.04755267
## 14      P3_strength       0.1635943           0.304241601  0.19212574
## 15 Cz_participation       0.1632078          -0.037513408 -0.40419767
```

This combined analysis links model-driven importance (SHAP) with neurobiological directionality and effect size, enabling clinically interpretable conclusions about Alzheimer's disease–related network alterations.

---

5. Neuroscientific interpretation and channel-level narrative reporting

Finally, we translate model outputs into **neuroscientifically interpretable, paper-style statements** that summarize how specific brain regions contribute to Alzheimer's disease classification.

For each influential EEG channel, the interpretation integrates the following complementary elements:

- **Direction of change**, quantified as the mean difference between Alzheimer's disease (AD) patients and healthy controls (HC)
- **Effect size**, measured using Cohen's d, reflecting the magnitude of group-level differences
- **Channel importance rank**, derived from aggregated SHAP values, indicating each channel's relative contribution to model predictions

This integrated reporting framework enables clinically meaningful interpretation of machine learning results, such as identifying **altered network efficiency**, **disrupted hub roles**, or **abnormal cross-regional integration** in specific brain regions associated with AD.

To generate clear and readable scientific narratives, EEG channels are mapped to their approximate **anatomical regions**, and channel-level statements are constructed by jointly considering:

- channel-level SHAP importance ranking
- the most influential node-level network feature(s) within each channel
- the direction of AD–HC differences
- corresponding Cohen's d effect sizes

Together, these narrative summaries bridge **statistical learning outputs** and **neuroscientific interpretation**, facilitating transparent reporting of region-specific network alterations related to Alzheimer's disease.

```
region_label <- function(ch) {
  if (ch %in% c("Fp1","Fp2")) return("frontal pole / prefrontal region")
  if (ch %in% c("F7","F3","Fz","F4","F8")) return("frontal region")
  if (ch %in% c("C3","Cz","C4")) return("central (sensorimotor) region")
  if (ch %in% c("T3","T4","T5")) return("temporal region")
  if (ch %in% c("P3","Pz","P4")) return("parietal (posterior) region")
  "brain region"
}

metric_explain <- function(metric_short) {
```

```
  if (metric_short == "local_eff") return("local information integration / propagation efficiency (local
  if (metric_short == "strength") return("overall connectivity intensity (node strength)")
  if (metric_short == "participation") return("cross-community connectivity (participation coefficient)
  if (metric_short == "closeness") return("global reachability / propagation distance (closeness)")
  if (metric_short == "betweenness") return("bridge role in network communication (betweenness)")
  if (metric_short == "clust") return("local clustering (clustering coefficient)")
  "network metric"
}

direction_text <- function(delta) {
  if (is.na(delta)) return("direction is unknown")
  if (delta < 0) return("is lower in AD than HC (AD < HC)")
  if (delta > 0) return("is higher in AD than HC (AD > HC)")
  "shows no difference"
}

channel_rank <- rank(-channel_importance, ties.method = "min")
top_channels <- names(head(channel_importance, 8))

make_channel_sentences <- function(ch, top_n_feats = 2) {
  feats_ch <- grep(paste0("^", ch, "_"), names(shap_imp), value = TRUE)
  feats_ch <- feats_ch[order(shap_imp[feats_ch], decreasing = TRUE)]
  top_feats <- head(feats_ch, top_n_feats)

  out <- c()
  for (ft in top_feats) {
    metric_short <- sub(paste0("^", ch, "_"), "", ft)
    delta <- diff_ad_minus_hc[ft]
    d_eff <- cohen_d[ft]

    out <- c(out,
      paste0(
        "The ", region_label(ch), " (", ch, ") shows that **", metric_short, "** (",
        metric_explain(metric_short), ") ",
        direction_text(delta), ". ",
        "This channel ranks **#", channel_rank[ch], "** in overall model contribution. ",
        "Effect size (Cohen's d) = ", sprintf("%.3f", d_eff), "."
      )
    )
  }
  out
}

cat("## Key Channel-Level Findings\n\n")
```

```
## ## Key Channel-Level Findings
```

```
for (ch in top_channels) {
  cat("### ", region_label(ch), " (", ch, ")\n", sep = "")

  lines <- make_channel_sentences(ch, top_n_feats = 2)

  for (ln in lines) {
```

```
    wrapped <- strwrap(ln, width = 80)
    cat("- ", wrapped[1], "\n", sep = "")
    if (length(wrapped) > 1) {
      for (w in wrapped[-1]) {
        cat("  ", w, "\n", sep = "")
      }
    }
  }

  cat("\n")
}
```

```
## ### central (sensorimotor) region (C3)
## - The central (sensorimotor) region (C3) shows that **betweenness** (bridge role
##   in network communication (betweenness)) is higher in AD than HC (AD > HC). This
##   channel ranks **#1** in overall model contribution. Effect size (Cohen's d) =
##   0.433.
## - The central (sensorimotor) region (C3) shows that **strength** (overall
##   connectivity intensity (node strength)) is higher in AD than HC (AD > HC). This
##   channel ranks **#1** in overall model contribution. Effect size (Cohen's d) =
##   0.871.
##
## ### central (sensorimotor) region (Cz)
## - The central (sensorimotor) region (Cz) shows that **betweenness** (bridge role
##   in network communication (betweenness)) is higher in AD than HC (AD > HC). This
##   channel ranks **#2** in overall model contribution. Effect size (Cohen's d) =
##   0.183.
## - The central (sensorimotor) region (Cz) shows that **participation**
##   (cross-community connectivity (participation coefficient)) is lower in AD than
##   HC (AD < HC). This channel ranks **#2** in overall model contribution. Effect
##   size (Cohen's d) = -0.404.
##
## ### parietal (posterior) region (P4)
## - The parietal (posterior) region (P4) shows that **closeness** (global
##   reachability / propagation distance (closeness)) is higher in AD than HC (AD >
##   HC). This channel ranks **#3** in overall model contribution. Effect size
##   (Cohen's d) = 0.736.
## - The parietal (posterior) region (P4) shows that **strength** (overall
##   connectivity intensity (node strength)) is higher in AD than HC (AD > HC). This
##   channel ranks **#3** in overall model contribution. Effect size (Cohen's d) =
##   0.728.
##
## ### frontal region (F7)
## - The frontal region (F7) shows that **betweenness** (bridge role in network
##   communication (betweenness)) is lower in AD than HC (AD < HC). This channel
##   ranks **#4** in overall model contribution. Effect size (Cohen's d) = -0.149.
## - The frontal region (F7) shows that **strength** (overall connectivity intensity
##   (node strength)) is higher in AD than HC (AD > HC). This channel ranks **#4**
##   in overall model contribution. Effect size (Cohen's d) = 0.413.
##
## ### frontal region (F3)
## - The frontal region (F3) shows that **strength** (overall connectivity intensity
##   (node strength)) is higher in AD than HC (AD > HC). This channel ranks **#5**
```

```
##   in overall model contribution. Effect size (Cohen's d) = 0.559.
## - The frontal region (F3) shows that **betweenness** (bridge role in network
##   communication (betweenness)) is higher in AD than HC (AD > HC). This channel
##   ranks **#5** in overall model contribution. Effect size (Cohen's d) = 0.048.
##
## ### parietal (posterior) region (P3)
## - The parietal (posterior) region (P3) shows that **strength** (overall
##   connectivity intensity (node strength)) is higher in AD than HC (AD > HC). This
##   channel ranks **#6** in overall model contribution. Effect size (Cohen's d) =
##   0.192.
## - The parietal (posterior) region (P3) shows that **participation**
##   (cross-community connectivity (participation coefficient)) is lower in AD than
##   HC (AD < HC). This channel ranks **#6** in overall model contribution. Effect
##   size (Cohen's d) = -0.221.
##
## ### frontal pole / prefrontal region (Fp2)
## - The frontal pole / prefrontal region (Fp2) shows that **betweenness** (bridge
##   role in network communication (betweenness)) is higher in AD than HC (AD > HC).
##   This channel ranks **#7** in overall model contribution. Effect size (Cohen's
##   d) = 0.090.
## - The frontal pole / prefrontal region (Fp2) shows that **closeness** (global
##   reachability / propagation distance (closeness)) is higher in AD than HC (AD >
##   HC). This channel ranks **#7** in overall model contribution. Effect size
##   (Cohen's d) = 0.561.
##
## ### frontal region (F8)
## - The frontal region (F8) shows that **betweenness** (bridge role in network
##   communication (betweenness)) is lower in AD than HC (AD < HC). This channel
##   ranks **#8** in overall model contribution. Effect size (Cohen's d) = -0.234.
## - The frontal region (F8) shows that **strength** (overall connectivity intensity
##   (node strength)) is higher in AD than HC (AD > HC). This channel ranks **#8**
##   in overall model contribution. Effect size (Cohen's d) = 0.323.
```

Interpretation note Because we use |corr| as edge weights, the reported "increase/decrease" primarily reflects **magnitude changes of functional coupling**, not whether connectivity becomes more positive or more negative.

6. Results

This pipeline provides a practical balance between predictive performance and interpretability:

1. **Network-based feature engineering** converts raw multichannel EEG into graph-derived connectivity descriptors that can be compared across windows/segments.
2. **Group-aware splitting** reduces optimistic bias by keeping windows from the same group separate between training and testing, thereby mitigating autocorrelation and segment leakage.
3. **SHAP aggregation** provides two complementary interpretability views:
    - *Which channels matter most?* (channel ranking)
    - *Which network properties matter most?* (metric ranking)

The analysis can be reviewed from the text outputs generated by the R code above.