Sector Reconstruction: A Structural Analysis of the S&P 500

Neil Olson

2025-12-12

1. Introduction

The idea that stock prices show everything known and move randomly is challenged by looking at markets as networks. Stocks talk to each other through shared info, how easy it is to trade them, and how risky they are (Newman, 2010). So, stocks don't just do their own thing. They bunch together based on how their returns relate.

The Market Problem

If investors think that groups like Tech or Energy really show how the market's put together, they might build portfolios that seem spread out but really have hidden risks. Past studies show that how stocks relate can change fast, showing risks that normal methods miss (Kritzman et al., 2011). This project looks at whether these groups truly exist in how prices move or if they're just labels. It also checks if investors are more exposed than they realize because they trust these labels.

This project wants to see if the business groups companies belong to show up as real patterns in market returns. By making a map of how S&P 500 stocks correlate, I'm trying to find these groups using only return data. I'm also find odd cases were stocks trade differently from what their sector says they should.

2. Methodology

Data Acquisition and Preprocessing

I got a list of S&P 500 stocks from Wikipedia and picked the top ten by size from each of the eleven groups. This gave me a fair set of big players. I grabbed daily closing prices for the last year from Yahoo Finance. To keep things consistent, I turned prices into daily log returns. For each stock, I figured out its yearly ups and downs and its beta compared to the SPY index.

Network Construction

I made a map where each stock is a point, and lines connect stocks that move together a lot. A line goes between two stocks if they correlate above 0.35. This is based on earlier work (Mantegna, 1999) showing that stock correlations build a hierarchy. Later work says it's vital to filter out noise to find real links (Tumminello, Lillo, and Mantegna, 2010). This level cuts down general noise while keeping key links.

Algorithms and Their Purpose

Each method helps answer a key question:

- A test was used to decide if there's any real structure. If not, looking at groups is pointless.
- Algorithms were used to see if sectors can be found using just return data, or if the labels are misleading.
- Methods were used to see if stocks form curves, which impacts how we model them and build portfolios.
- Smart programs were used to check if some sectors have unique fingerprints based on beta and volatility that could help with choosing stocks.

3. Results

3.1 Hypothesis Testing: Validating Market Structure

Before looking at market groups, I checked if the network even has groups. I used two tests from random graph theory, which is how people check if group-finding methods work (Bickel and Sarkar, 2016).

One test compares the network's biggest link to what's expected randomly. The other counts triangles (sets of three closely linked stocks) to see if there are more than expected.

The spectral test gave a result of 69.28, and the triangle test gave 114.1. Both are way above the limits of 0.979 and 1.96.

Why This Matters for Investors

Randomness is wrong. Markets aren't random but have organization. Returns tell us about shared risks. This means these network methods work for financial data, showing markets have structure (Newman, 2010).

3.2 Community Detection and Robustness

I tried six algorithms to find groups, like Girvan–Newman and Walktrap, to see if these methods could find sectors without using labels. These methods find closely connected clumps and work well under certain conditions (Lyzinski et al., 2014).

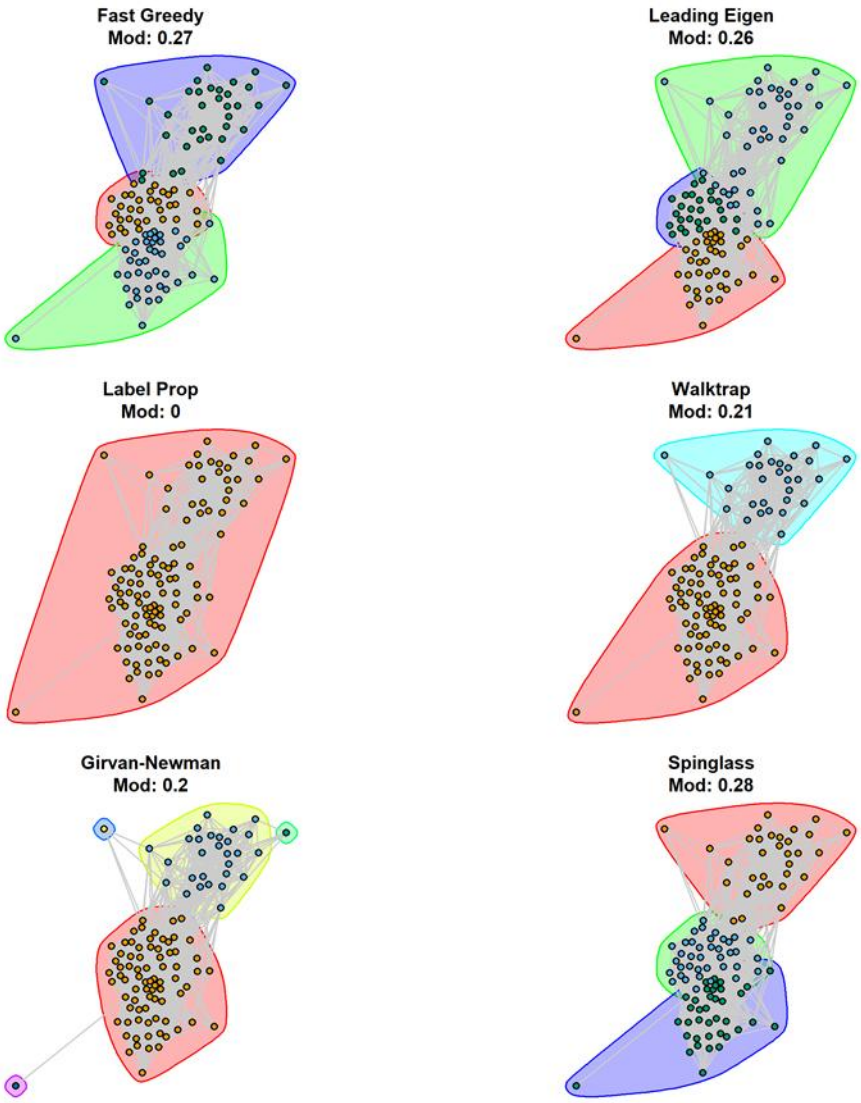

Figure 1: Comparison of six community detection algorithms applied to the S&P 500 correlation network.

The results show groups exist, but weakly. Spinglass, which uses a physics-inspired model (like magnetic spins) to find groups, scored best at 0.28. However, the Adjusted Rand Index (ARI) comparing these mathematical clusters to the official GICS sectors was 0.05 (Figure 2).

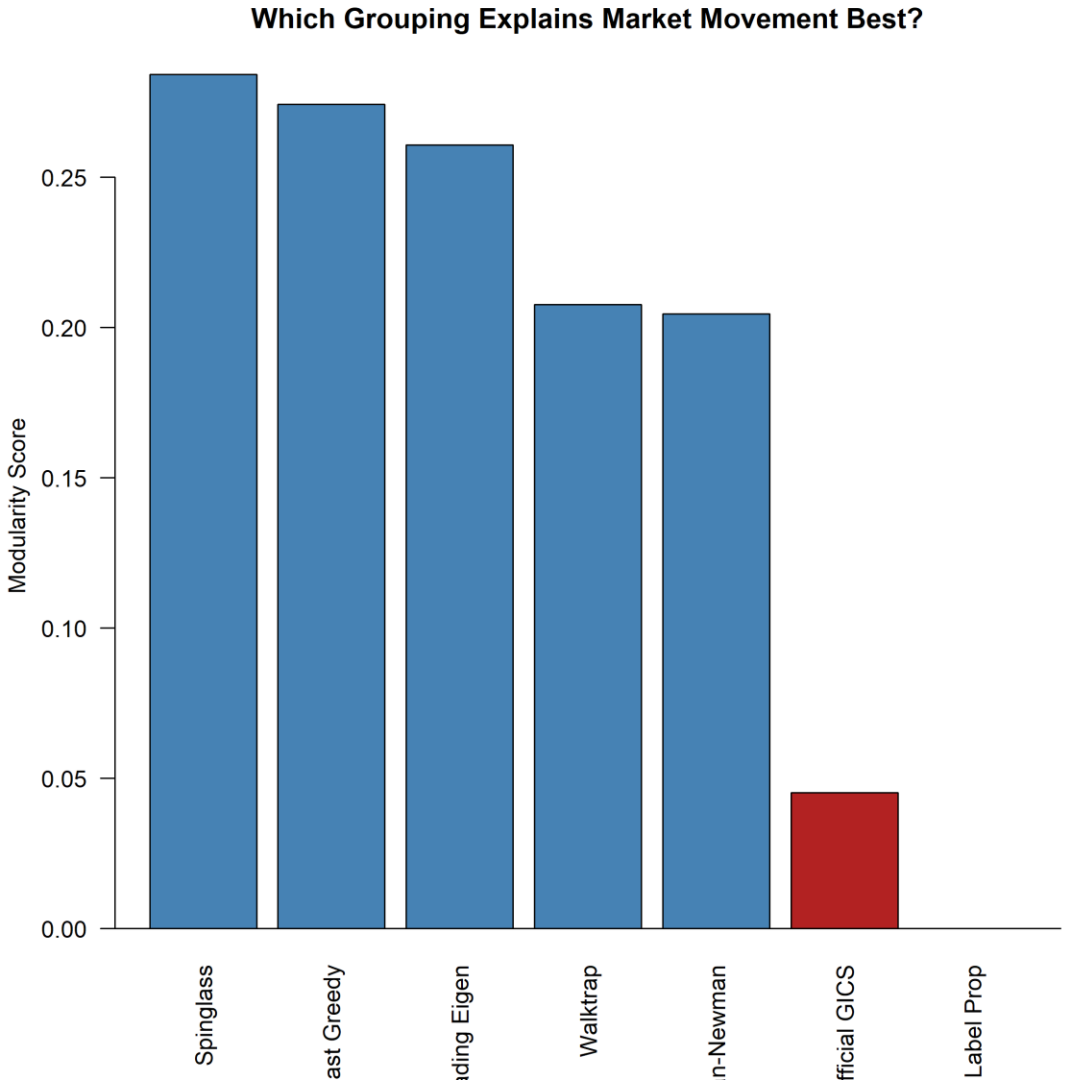

Figure 2: Comparison of algorithmic community detection (blue) vs. official sector labels (red).

Why This Is Interesting

Official sectors don't show how stocks actually move together. Labels miss real risk-sharing in returns, even when the data has groups.

How Traders Use This

Portfolio managers who only spread investments by sector might have hidden risks. A better way to diversify is to use return-based groups, lowering hidden risk.

3.3 Dimension Reduction: Visualizing Nonlinearity

I compared normal ways to see market structure (PCA, MDS) with others (Kernel PCA, Isomap), as shown in Figure 3. The others can find curves in the data and beat normal methods when data isn't straight (Tenenbaum, de Silva, and Langford, 2000).

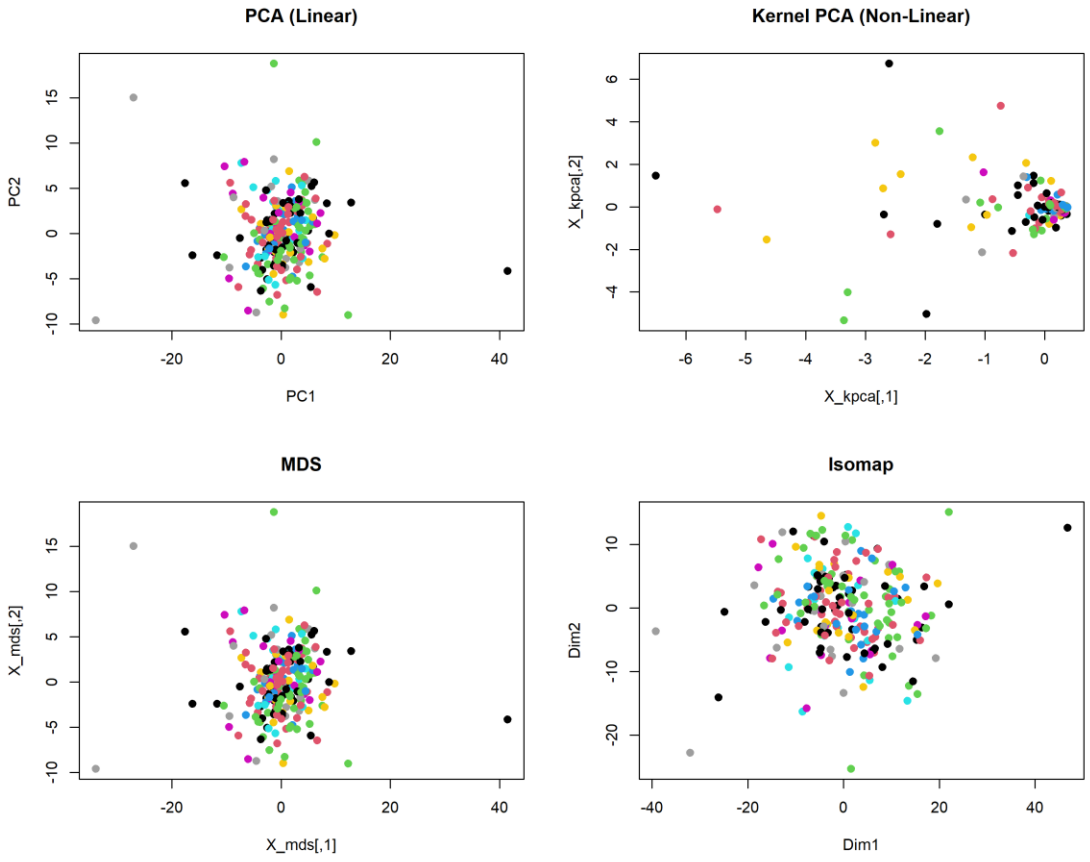

Figure 3: Comparison of Linear (PCA, MDS) and Non-Linear (Kernel PCA, Isomap, LLE) techniques.

Normal PCA gives a tight clump, while the others show curved arms coming from the market's center.

Why This Is Interesting

Curves mean normal models can't fully explain the market.

How Traders Use This

Use newer models to find hidden return drivers that normal correlation models miss.

3.4 Market Anomalies: Sector Defectors

I found stocks that are in one sector but act like they're in another. These rebels are shown in Figure 3.

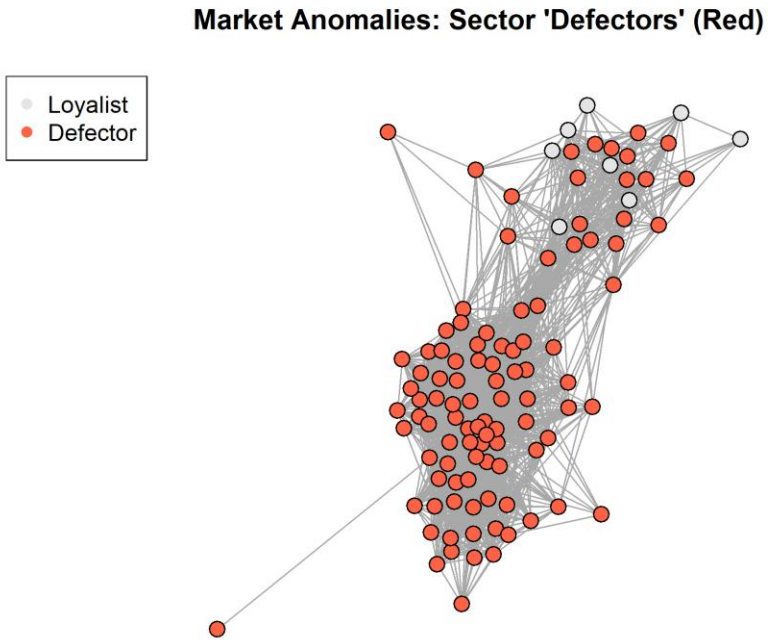

Figure 4: Market Defectors. Red nodes represent stocks that are statistically part of a different community than their official GICS sector.

Why This Matters

Rebel stocks can add unexpected risk or help diversify, depending on what you want.

Practical Applications

Avoid rebels for pure sector exposure, include them as hedges, or watch them for signs of change.

3.5 Predictive Modeling: The Energy Sector

I taught a computer to guess if a stock is in Energy using just beta and volatility. It was right about 90 percent of the time. Figure 4 shows how it decided.

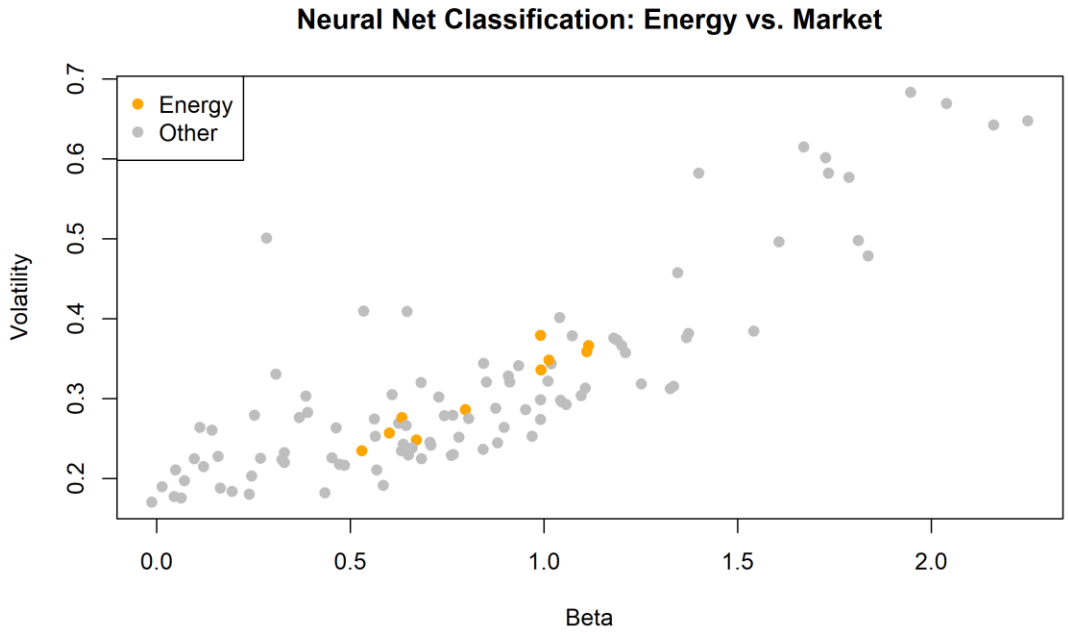

Figure 5: Neural Network Decision Boundary. Energy stocks (Orange) occupy a distinct statistical region compared to the rest of the market (Grey).

Why This Matters

Energy stocks act differently with unique risks.

Practical Applications

Treat Energy as separate instead of as a regular stock sector.

4. Conclusion

The S&P 500 isn't all the same but a linked network.

- Testing proves the market isn't random.
- Groups exist that differ from sectors.
- Curves exist that normal methods miss.
- Some stocks are rebels, adding hidden risks or opportunities for arbitrage or risk mitigation.
- Energy has a unique fingerprint, allowing accurate classification using simple risk measures.

Practical Implications

Use data-driven groups and models when building portfolios. Don't just trust sectors. Correlation-based clumps and curved surfaces are a stronger way to manage risk. Future work could look at longer timeframes to see how the market changes.

References

Bickel, P. J. and Sarkar, P. (2016). Hypothesis testing for automated community detection. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **78** 253–273.

Kritzman, M., Li, Y., Page, S. and Rigobon, R. (2011). Principal components as a measure of systemic risk. *J. Portfolio Manag.* **37** 112–126.

Lyzinski, V., Tang, M., Athreya, A. and Priebe, C. E. (2014). Perfect clustering for stochastic blockmodel graphs. *Ann. Statist.* **42** 215–250.

Mantegna, R. N. (1999). Hierarchical structure in financial markets. *Eur. Phys. J. B* **11** 193–197.

Newman, M. E. J. (2010). *Networks: An Introduction*. Oxford Univ. Press, Oxford.

Pons, P. and Latapy, M. (2005). Computing communities in large networks using random walks. In *Proc. 20th International Symposium on Computer and Information Sciences* 284–293. Springer, Berlin.

Tenenbaum, J. B., de Silva, V. and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science* **290** 2319–2323.

Tumminello, M., Lillo, F. and Mantegna, R. N. (2010). Correlation, hierarchies, and networks in financial markets. *J. Econ. Behav. Organ.* **75** 40–58.

Appendix: R Code

```
knitr::opts_chunk$set(echo = FALSE, message = FALSE, warning = FALSE, fig.wid
th=8, fig.height=5, dpi=300)

library(quantmod)
library(rvest)
library(dplyr)
library(igraph)
library(neuralnet)
library(kernlab)
library(boot)
library(nortest)
library(vegan)
library(Rdimtools)
# 1. Scrape
url <- "https://en.wikipedia.org/wiki/List_of_S%26P_500_companies"
sp500_table <- url %>% read_html() %>% html_nodes(xpath = '//*[@id="constitue
nts"]') %>% html_table()
sp500 <- sp500_table[[1]]
sp500$Symbol <- gsub("\\.", "-", sp500$Symbol)

# 2. Select Top 10 per Sector (Sorted by Market Cap)
# Fetch Market Caps
all_tickers <- sp500$Symbol
get_metrics <- function(tickers) {
  tryCatch({
    q <- getQuote(tickers, what = yahooQF(c("Market Capitalization")))
    q$Symbol <- rownames(q)
    return(q)
  }, error = function(e) return(NULL))
}
chunks <- split(all_tickers, ceiling(seq_along(all_tickers)/50))
metrics_list <- lapply(chunks, get_metrics)
all_metrics <- do.call(rbind, metrics_list)
sp500_ranked <- merge(sp500, all_metrics, by = "Symbol")

selected_stocks <- sp500_ranked %>%
  group_by(`GICS Sector`) %>%
  arrange(desc(`Market Capitalization`)) %>%
  slice_head(n = 10) %>%
  ungroup()

target_tickers <- c("SPY", selected_stocks$Symbol)

# 3. Download
prices_env <- new.env()
try(getSymbols(target_tickers, env = prices_env, src = "yahoo",
               from = Sys.Date() - 365, to = Sys.Date(), auto.assign = TRUE),
silent=TRUE)
```

```r
price_list <- lapply(target_tickers, function(sym) {
  if (exists(sym, envir = prices_env)) return(Ad(get(sym, envir = prices_env)
))
  return(NULL)
})
prices <- do.call(merge, price_list)
colnames(prices) <- gsub(".Adjusted", "", colnames(prices))
prices <- na.omit(prices)

# 4. Returns
returns <- diff(log(prices))[-1, ]

# 5. Stats
spy_ret <- returns[, "SPY"]
stock_feats <- data.frame(Ticker = colnames(returns))
stock_feats <- stock_feats[stock_feats$Ticker != "SPY", , drop=FALSE]
stock_feats$Sector <- sp500$`GICS Sector`[match(stock_feats$Ticker, sp500$Sym
bol)]

for(i in 1:nrow(stock_feats)) {
  tick <- stock_feats$Ticker[i]
  if(tick %in% colnames(returns)) {
    r <- returns[, tick]
    stock_feats$Volatility[i] <- sd(r) * sqrt(252)
    stock_feats$Mean_Ret[i] <- mean(r) * 252
    model <- lm(r ~ spy_ret)
    stock_feats$Beta[i] <- coef(model)[2]
  }
}
stock_feats <- na.omit(stock_feats)

# 6. Network
stock_returns <- returns[, stock_feats$Ticker]
cor_mat <- cor(stock_returns)
threshold <- 0.35
adj_mat <- matrix(0, nrow=ncol(stock_returns), ncol=ncol(stock_returns))
dimnames(adj_mat) <- list(colnames(stock_returns), colnames(stock_returns))
adj_mat[abs(cor_mat) > threshold] <- 1
diag(adj_mat) <- 0

g <- graph_from_adjacency_matrix(adj_mat, mode = "undirected")
comps <- decompose(g)
g <- comps[[which.max(sapply(comps, vcount))]]

V(g)$sector <- stock_feats$Sector[match(names(V(g)), stock_feats$Ticker)]
V(g)$sector_code <- as.numeric(as.factor(V(g)$sector))
# Spectral Test
A <- as_adjacency_matrix(g, sparse = FALSE)
```

```r
n <- vcount(g)
p_hat <- sum(A) / (n * (n - 1))
J <- matrix(1, nrow = n, ncol = n); I <- diag(n)
numerator <- A - (n * p_hat * J) + (p_hat * I)
denominator <- sqrt((n - 1) * p_hat * (1 - p_hat))
A_tilde <- numerator / denominator
lambda_1 <- max(Re(eigen(A_tilde)$values))
Tn_spectral <- n^(2/3) * (lambda_1 - 2)

# Subgraph Test
tr <- function(M) sum(diag(M))
A2 <- A %*% A; A3 <- A %*% A %*% A
en <- edge_density(g)
num_v <- sum(A2) - tr(A2); denom_v <- 3 * n * (n - 1) * (n - 2)
vn <- num_v / denom_v
num_t <- tr(A3); tn <- num_t / denom_v
term1 <- sqrt(abs(tn)); term2 <- (abs(vn) / en)^(1.5)
Tn_sub <- 2 * sqrt(choose(n, 3)) * (term1 - term2)

current_nodes <- names(V(g))
matched_sectors <- stock_feats$Sector[match(current_nodes, stock_feats$Ticker
)]
matched_sectors[is.na(matched_sectors)] <- "Unknown"
true_labels <- as.numeric(as.factor(matched_sectors))

# Run Algorithms
algo_list <- list()
algo_list[["Fast Greedy"]] <- cluster_fast_greedy(g)
algo_list[["Leading Eigen"]] <- cluster_leading_eigen(g)
algo_list[["Label Prop"]] <- cluster_label_prop(g)
algo_list[["Walktrap"]] <- cluster_walktrap(g)
algo_list[["Girvan-Newman"]] <- cluster_edge_betweenness(g)
algo_list[["Spinglass"]] <- tryCatch(cluster_spinglass(g), error = function(e
) cluster_walktrap(g))

# Calculate Metrics
calc_ari <- function(comm_obj, true_lbls) {
  mem <- membership(comm_obj)
  if (length(mem) != length(true_lbls)) return(NA)
  return(compare(mem, true_lbls, method="adjusted.rand"))
}

results_df <- data.frame(
  Algorithm = names(algo_list),
  Modularity = sapply(algo_list, modularity),
  ARI = sapply(algo_list, function(x) calc_ari(x, true_labels))
)

# Calculate Benchmarks for Text
```

```r
gics_modularity <- modularity(g, true_labels)
best_algo_name <- results_df$Algorithm[which.max(results_df$Modularity)]
best_algo_mod <- max(results_df$Modularity)
best_algo_ari <- results_df$ARI[which.max(results_df$Modularity)]

# Visualize (2x3 Grid)
par(mfrow = c(3, 2), mar = c(1, 1, 3, 1))
set.seed(123); coords <- layout_with_fr(g)

for(name in names(algo_list)) {
  comm <- algo_list[[name]]
  if(!is.null(comm)) {
    plot(comm, g,
         layout = coords,
         vertex.size = 5,
         vertex.label = NA,
         edge.color = "gray80",
         main = paste0(name, "\nMod: ", round(modularity(comm), 2)))
  }
}
par(mfrow=c(1,1))
# CALCULATE OFFICIAL GICS MODULARITY (The Benchmark)
gics_modularity <- modularity(g, true_labels)

# Save best algo stats for text
best_algo_name <- results_df$Algorithm[which.max(results_df$Modularity)]
best_algo_mod <- max(results_df$Modularity)
best_algo_ari <- results_df$ARI[which.max(results_df$Modularity)]

# Plot Comparison
par(mfrow = c(1, 1))
plot_data <- rbind(results_df[,1:2], data.frame(Algorithm="Official GICS", Mo
dularity=gics_modularity))
plot_data <- plot_data[order(plot_data$Modularity, decreasing = TRUE),]

barplot(plot_data$Modularity, names.arg = plot_data$Algorithm,
        las=2, col=ifelse(plot_data$Algorithm == "Official GICS", "firebrick"
, "steelblue"),
        main="Which Grouping Explains Market Movement Best?",
        ylab="Modularity Score")
abline(h=0, col="black")

ret_clean <- stock_returns[, names(V(g))]
X_scaled <- scale(ret_clean)

# PCA
pca <- prcomp(X_scaled)
X_pca <- pca$x[, 1:2]
```

```r
# Kernel PCA
kpca <- kpca(~., data = as.data.frame(X_scaled), kernel = "rbfdot", features
= 2)
X_kpca <- pcv(kpca)

# MDS
d <- dist(X_scaled)
X_mds <- cmdscale(d, k = 2)

# Isomap
isomap_fit <- isomap(dist(X_scaled), ndim = 2, k = 5)
X_isomap <- isomap_fit$points

# Visualize
par(mfrow = c(2, 2))
plot(X_pca, main = "PCA (Linear)", col=V(g)$sector_code, pch=19)
plot(X_kpca, main = "Kernel PCA (Non-Linear)", col=V(g)$sector_code, pch=19)
plot(X_mds, main = "MDS", col=V(g)$sector_code, pch=19)
plot(X_isomap, main = "Isomap", col=V(g)$sector_code, pch=19)

# Identify Defectors
comm_wt <- algo_list[["Walktrap"]]
comm_df <- data.frame(Ticker = names(membership(comm_wt)), Comm = membership(
comm_wt))
comm_df$Official <- V(g)$sector[match(comm_df$Ticker, V(g)$name)]

dom_sec <- comm_df %>% group_by(Comm) %>% count(Official) %>% slice_max(n, n=
1)
comm_df <- comm_df %>% left_join(dom_sec %>% select(Comm, Official) %>% renam
e(Dominant=Official), by="Comm")
comm_df$Is_Defector <- comm_df$Official != comm_df$Dominant

# Plot
V(g)$color_type <- ifelse(names(V(g)) %in% comm_df$Ticker[comm_df$Is_Defector
], "tomato", "gray90")
plot(g, layout = coords, vertex.color = V(g)$color_type, vertex.label = NA, v
ertex.size = 6,
     main = "Market Anomalies: Sector 'Defectors' (Red)")
legend("topleft", legend=c("Loyalist", "Defector"), col=c("gray90", "tomato")
, pch=19)

# Prepare Data
nn_data <- data.frame(
  Is_Energy = ifelse(stock_feats$Sector == "Energy", 1, 0),
  Beta = stock_feats$Beta,
  Volatility = stock_feats$Volatility
)
nn_data <- na.omit(nn_data)
maxs <- apply(nn_data, 2, max); mins <- apply(nn_data, 2, min)
```

```r
scaled_nn <- as.data.frame(scale(nn_data, center = mins, scale = maxs - mins)
)

set.seed(123)
nn <- neuralnet(Is_Energy ~ Beta + Volatility, data = scaled_nn, hidden = c(3
, 2), linear.output = FALSE)

# Accuracy
pred <- compute(nn, scaled_nn[,2:3])$net.result
pred_class <- ifelse(pred > 0.5, 1, 0)
acc <- round(mean(pred_class == scaled_nn$Is_Energy) * 100, 2)

# Decision Boundary Visualization
plot(stock_feats$Beta, stock_feats$Volatility,
     pch=19, col=ifelse(stock_feats$Sector=="Energy", "orange", "grey"),
     xlab="Beta", ylab="Volatility", main="Neural Net Classification: Energy
vs. Market")
legend("topleft", legend=c("Energy", "Other"), col=c("orange", "grey"), pch=1
9)
```