Take-Home Exercise 2

Marga Thura

2025-05-13

# Load the necessary libraries

pacman::p\_load(tidyverse, jsonlite, SmartEDA, tidygraph, ggraph, packcircles, stringr, igraph, visNetwork, treemap)

# Read the data

# Read JSON file  
mc1\_data <- fromJSON("MC1\_release/MC1\_graph.json")

# Overview the data

glimpse(mc1\_data)

List of 5  
 $ directed : logi TRUE  
 $ multigraph: logi TRUE  
 $ graph :List of 2  
 ..$ node\_default: Named list()  
 ..$ edge\_default: Named list()  
 $ nodes :'data.frame': 17412 obs. of 10 variables:  
 ..$ Node Type : chr [1:17412] "Song" "Person" "Person" "Person" ...  
 ..$ name : chr [1:17412] "Breaking These Chains" "Carlos Duffy" "Min Qin" "Xiuying Xie" ...  
 ..$ single : logi [1:17412] TRUE NA NA NA NA FALSE ...  
 ..$ release\_date : chr [1:17412] "2017" NA NA NA ...  
 ..$ genre : chr [1:17412] "Oceanus Folk" NA NA NA ...  
 ..$ notable : logi [1:17412] TRUE NA NA NA NA TRUE ...  
 ..$ id : int [1:17412] 0 1 2 3 4 5 6 7 8 9 ...  
 ..$ written\_date : chr [1:17412] NA NA NA NA ...  
 ..$ stage\_name : chr [1:17412] NA NA NA NA ...  
 ..$ notoriety\_date: chr [1:17412] NA NA NA NA ...  
 $ links :'data.frame': 37857 obs. of 4 variables:  
 ..$ Edge Type: chr [1:37857] "InterpolatesFrom" "RecordedBy" "PerformerOf" "ComposerOf" ...  
 ..$ source : int [1:37857] 0 0 1 1 2 2 3 5 5 5 ...  
 ..$ target : int [1:37857] 1841 4 0 16180 0 16180 0 5088 14332 11677 ...  
 ..$ key : int [1:37857] 0 0 0 0 0 0 0 0 0 0 ...

# Inspect structure

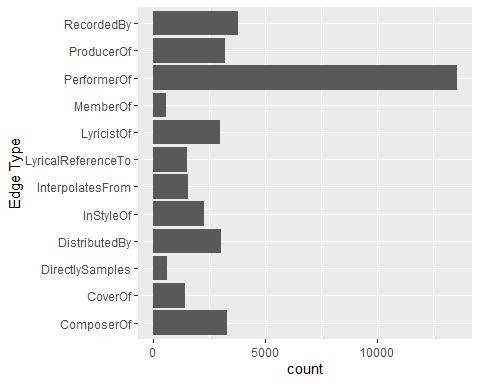
str(mc1\_data, max.level=1)

List of 5  
 $ directed : logi TRUE  
 $ multigraph: logi TRUE  
 $ graph :List of 2  
 $ nodes :'data.frame': 17412 obs. of 10 variables:  
 $ links :'data.frame': 37857 obs. of 4 variables:

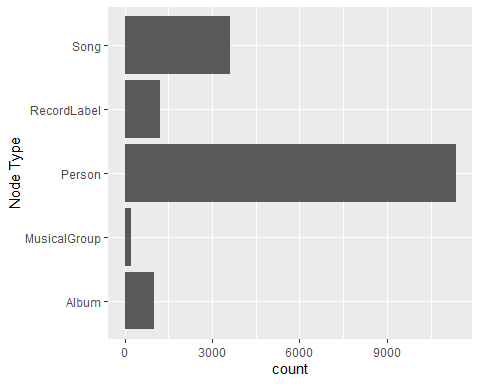
# Extract and Inspect

nodes\_tbl <- as\_tibble(mc1\_data$nodes)  
  
edges\_tbl <- as\_tibble(mc1\_data$links)

ggplot(data = edges\_tbl,   
 aes(y = `Edge Type`)) +  
 geom\_bar()



ggplot(data = nodes\_tbl,  
 aes(y = `Node Type`)) +  
 geom\_bar()



# Creating knowledge graph

Mapping from node id to row index. Ensure each id from your node list is mapped to the correct row number.

id\_map <- tibble(id = nodes\_tbl$id,  
 index = seq\_len(  
 nrow(nodes\_tbl)))

edges\_tbl <- edges\_tbl %>%  
 left\_join(id\_map, by = c("source" = "id")) %>%  
 rename(from = index) %>%  
 left\_join(id\_map, by = c("target" = "id")) %>%  
 rename(to = index)

# Remove the NA

edges\_tbl <- edges\_tbl %>%  
 filter(!is.na(from), !is.na(to))

# Creating the graph

graph <- tbl\_graph(nodes = nodes\_tbl,  
 edges = edges\_tbl,  
 directed = mc1\_data$directed)

# Visualising the knowledge graph

set.seed(1234)

# Visualising the whole graph

ggraph(graph, layout = "fr") +  
 geom\_edge\_link(alpha = 0.3,  
 colour = "gray") +  
 geom\_node\_point(aes(color = `Node Type`),  
 size = 4) +  
 geom\_node\_text(aes(label = name),  
 repel = TRUE,  
 size = 2.5) +  
 theme\_void()

# Filter edges to only MemberOf

graph\_memberof <- graph %>%  
 activate(edges) %>%   
 filter(`Edge Type` == "MemberOf")

# Extract only connected nodes (used in the edges)

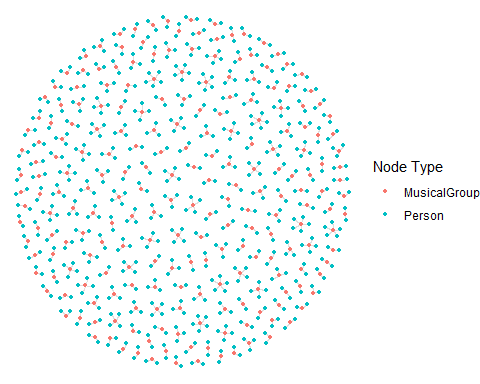
used\_nodes\_indices <- graph\_memberof %>%  
 activate(edges) %>%  
 as\_tibble() %>%  
 select(from, to) %>%  
 unlist() %>%  
 unique()

# Keep only those nodes

graph\_memberof <- graph\_memberof %>%  
 activate(nodes) %>%  
 mutate(row\_id = row\_number()) %>%  
 filter(row\_id %in% used\_nodes\_indices) %>%  
 select(-row\_id)

# Plot the sub-graph

ggraph(graph\_memberof,  
 layout = "fr") +  
 geom\_edge\_link(alpha = 0.5,  
 colour = "gray") +  
 geom\_node\_point(aes(color = `Node Type`),  
 size = 1) +  
 geom\_node\_text(aes(label = name),  
 repel = TRUE,  
 size = 2.5) +  
 theme\_void()



# Task 1

# Part A: Who has Sailor Shift been most influenced by over time?

To understand how Sailor Shift’s musical style has evolved, we examined the influence relationships affecting her songs over time. We defined influence using five edge types from the dataset: InStyleOf, CoverOf, DirectlySamples, InterpolatesFrom, and LyricalReferenceTo.

We began by identifying all songs performed by Sailor Shift and organizing them chronologically using their release dates. This temporal ordering allowed us to assess changes in external influences throughout her career.

Next, we filtered the full graph to isolate influential edges targeting Sailor’s songs. These were joined with metadata to identify the artists, groups, or labels responsible for the original source of influence. To ensure accuracy, we also validated that the influencing node was a valid performer using song-to-performer relationships.

## A1. Convert & Validate Dates

# Check raw release\_date format  
nodes\_tbl %>%  
 select(name, release\_date) %>%  
 filter(!is.na(release\_date)) %>%  
 distinct(release\_date) %>%  
 arrange(release\_date)

# A tibble: 64 × 1  
 release\_date  
 <chr>   
 1 1975   
 2 1977   
 3 1979   
 4 1980   
 5 1981   
 6 1982   
 7 1983   
 8 1984   
 9 1985   
10 1986   
# ℹ 54 more rows

## A2. Check Sailor Shift’s song release date sorted chronologically

# 1. Find Sailor Shift's ID  
sailor\_id <- nodes\_tbl %>%  
 filter(name == "Sailor Shift") %>%  
 pull(id)  
  
# 2. Get all songs she performed  
sailor\_songs <- edges\_tbl %>%  
 filter(`Edge Type` == "PerformerOf", source == sailor\_id) %>%  
 pull(target)  
  
# 3. Retrieve and sort their release dates  
sailor\_songs\_tbl <- nodes\_tbl %>%  
 filter(id %in% sailor\_songs) %>%  
 select(id, name, release\_date) %>%  
 mutate(release\_year = as.numeric(release\_date)) %>%  
 arrange(release\_year)  
  
# 4. Print result  
print(sailor\_songs\_tbl)

# A tibble: 26 × 4  
 id name release\_date release\_year  
 <int> <chr> <chr> <dbl>  
 1 17272 Tidal Pop Waves 2028 2028  
 2 17279 High Tide Heartbeat 2028 2028  
 3 17280 Electric Eel Love 2028 2028  
 4 17281 Sun-Drenched Daydream 2028 2028  
 5 17282 Chord of the Deep 2028 2028  
 6 17273 Salty Dreams 2030 2030  
 7 17283 Heart of the Habitat 2030 2030  
 8 17284 Reef Rhythm 2030 2030  
 9 17285 Driftwood Lullaby 2030 2030  
10 17410 Seashell Serenade 2030 2030  
# ℹ 16 more rows

## A3. Influence Types

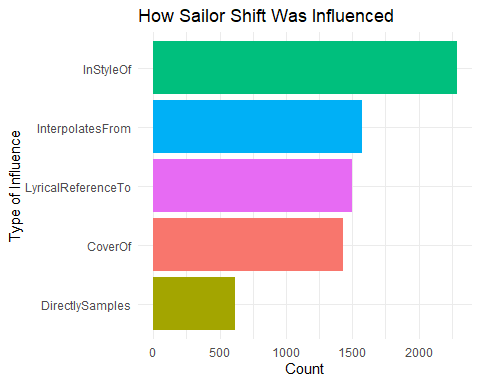
influence\_types <- c(  
 "InStyleOf",  
 "CoverOf",  
 "DirectlySamples",  
 "InterpolatesFrom",  
 "LyricalReferenceTo"  
)

### Build influence data table

influences <- edges\_tbl %>%  
 filter(`Edge Type` %in% influence\_types)

### Plot influence

influences %>%  
 count(`Edge Type`) %>%  
 ggplot(aes(x = reorder(`Edge Type`, n), y = n, fill = `Edge Type`)) +  
 geom\_col(show.legend = FALSE) +  
 coord\_flip() +  
 labs(title = "How Sailor Shift Was Influenced",  
 x = "Type of Influence", y = "Count") +  
 theme\_minimal()



## A4. Find the song released dates

library(dplyr)  
  
# 1. Define the years in which she released music  
years <- sort(unique(sailor\_songs\_tbl$release\_year))  
  
years

[1] 2028 2030 2031 2032 2034 2036 2038 2040

### Define influence\_types

library(dplyr)  
  
yearly\_top\_influencers <-   
 edges\_tbl %>%  
 # 1. keep only influence edges  
 filter(`Edge Type` %in% influence\_types) %>%  
 # 2. attach Sailor Shift song years (join on source → id)  
 inner\_join(  
 sailor\_songs\_tbl,  
 by = c("source" = "id")  
 ) %>%  
 # 3. bring in influencer name + node type  
 left\_join(  
 nodes\_tbl %>% select(id, influencer = name, type = `Node Type`),  
 by = c("target" = "id")  
 ) %>%  
 # 4. tally per year + influencer  
 count(release\_year, influencer, type, name = "count") %>%  
 # 5. pick the single top influencer each year  
 group\_by(release\_year) %>%  
 slice\_max(count, n = 1, with\_ties = FALSE) %>%  
 ungroup() %>%  
 arrange(release\_year)  
  
print(yearly\_top\_influencers)

# A tibble: 7 × 4  
 release\_year influencer type count  
 <dbl> <chr> <chr> <int>  
1 2028 Addicted to Your Heartache Album 1  
2 2030 Echoes of Forgotten Light Song 1  
3 2031 Dreamscape of Judgment Song 1  
4 2032 Parallel Memories Song 1  
5 2034 Divergent Memories Song 1  
6 2036 Coastal Whispers of Biscay Song 1  
7 2038 Altitude of Mistakes Song 1

### Build perf\_map

perf\_map <- edges\_tbl %>%   
 filter(`Edge Type` == "PerformerOf") %>%   
 select(influencer\_song = target,   
 performer\_id = source)

### Build yearly\_top\_influencers\_artists

yearly\_top\_influencers\_artists <-  
 edges\_tbl %>%  
 filter(`Edge Type` %in% influence\_types) %>%  
   
 # attach Sailor Shift’s song years  
 inner\_join(sailor\_songs\_tbl, by = c("source" = "id")) %>%  
   
 # attach the song→performer mapping  
 inner\_join(perf\_map, by = c("target" = "influencer\_song")) %>%  
   
 # get the performer’s name & node type  
 left\_join(  
 nodes\_tbl %>% select(id, influencer = name, type = `Node Type`),  
 by = c("performer\_id" = "id")  
 ) %>%  
   
 # only keep real performers (Person / Music Group / Record Label)  
 filter(type %in% c("Person", "Music Group", "Record Label")) %>%  
   
 # count per year + performer  
 count(release\_year, influencer, type, name = "count") %>%  
   
 # pick the top performer each year  
 group\_by(release\_year) %>%  
 slice\_max(count, n = 1, with\_ties = FALSE) %>%  
 ungroup() %>%  
 arrange(release\_year)  
  
print(yearly\_top\_influencers\_artists)

# A tibble: 7 × 4  
 release\_year influencer type count  
 <dbl> <chr> <chr> <int>  
1 2028 Jeremiah Love Person 1  
2 2030 Gang Shao Person 1  
3 2031 Sandra Burke Person 1  
4 2032 Guiying Ren Person 1  
5 2034 Joshua Taylor Person 1  
6 2036 Daniel Mccormick Person 1  
7 2038 Amico Luciani Person 1

To compare influence levels across different types (e.g., lyrical vs. stylistic), we computed normalized weights: rarer influence types were assigned higher scores to balance their impact in the analysis. For each year in which Sailor released music, we calculated a weighted influence score for every performer that influenced her that year.

## A5. Build weighted counts

# 1. Count how many times each Edge Type occurs  
base\_counts <- influences %>%  
 count(`Edge Type`, name = "n")  
  
# 2. Examine  
print(base\_counts)

# A tibble: 5 × 2  
 `Edge Type` n  
 <chr> <int>  
1 CoverOf 1429  
2 DirectlySamples 619  
3 InStyleOf 2289  
4 InterpolatesFrom 1574  
5 LyricalReferenceTo 1496

### Auto\_weights

# 3. Compute max frequency  
max\_n <- max(base\_counts$n)  
  
# 4. Build the named weight vector  
auto\_weights <- base\_counts %>%  
 mutate(weight = max\_n / n) %>%  
 select(`Edge Type`, weight) %>%  
 deframe()  
  
print(auto\_weights)

CoverOf DirectlySamples InStyleOf InterpolatesFrom   
 1.601819 3.697900 1.000000 1.454257   
LyricalReferenceTo   
 1.530080

### Build weights → weighted\_top\_by\_year\_auto

weighted\_top\_by\_year\_auto <- edges\_tbl %>%  
 filter(`Edge Type` %in% influence\_types) %>%  
 inner\_join(sailor\_songs\_tbl, by = c("source" = "id")) %>%  
 inner\_join(perf\_map, by = c("target" = "influencer\_song")) %>%  
 left\_join(  
 nodes\_tbl %>% select(id, performer = name, type = `Node Type`),  
 by = c("performer\_id" = "id")  
 ) %>%  
 filter(type %in% c("Person", "Music Group", "Record Label")) %>%  
 mutate(weight = recode(`Edge Type`, !!!auto\_weights)) %>%  
 group\_by(release\_year, performer, type) %>%  
 summarize(score = sum(weight), .groups = "drop") %>%  
 group\_by(release\_year) %>%  
 slice\_max(score, n = 1, with\_ties = FALSE) %>%  
 ungroup() %>%  
 arrange(release\_year)  
  
print(weighted\_top\_by\_year\_auto)

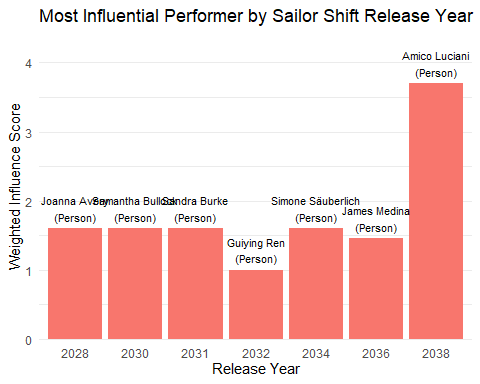
# A tibble: 7 × 4  
 release\_year performer type score  
 <dbl> <chr> <chr> <dbl>  
1 2028 Joanna Avery Person 1.60  
2 2030 Samantha Bullock Person 1.60  
3 2031 Sandra Burke Person 1.60  
4 2032 Guiying Ren Person 1   
5 2034 Simone Säuberlich Person 1.60  
6 2036 James Medina Person 1.45  
7 2038 Amico Luciani Person 3.70

## A6. Most Influential Performer by Sailor Shift Release Year

The resulting chart below shows the top influencer by year, accounting for both the frequency and type of influence. Each bar represents a release year in Sailor’s career, annotated with the name of her top influencer (person, group, or label) and the cumulative influence score.

This approach reveals both consistent sources of inspiration and shifts in influence throughout her discography, offering insight into how her sound may have been shaped by key collaborators or predecessors.

library(ggplot2)  
  
ggplot(weighted\_top\_by\_year\_auto,  
 aes(x = factor(release\_year), y = score, fill = type)) +  
 geom\_col(show.legend = FALSE) +  
 geom\_text(aes(label = paste0(performer, "\n(", type, ")")),  
 vjust = -0.3, size = 3) +  
 scale\_y\_continuous(expand = expansion(mult = c(0, 0.2))) +  
 labs(  
 title = "Most Influential Performer by Sailor Shift Release Year",  
 x = "Release Year",  
 y = "Weighted Influence Score"  
 ) +  
 theme\_minimal() +  
 theme(  
 axis.text.x = element\_text(angle = 0, vjust = 0.5),  
 panel.grid.major.x = element\_blank()  
 )



# Part B: Who has Sailor Shift collaborated with and directly or indirectly influenced?

To answer this question, we first identified all artists who collaborated with Sailor Shift, and then checked whether any of these collaborators were later influenced by her music either directly (1 hop) or indirectly (2 hops).

Next, we extracted all PerformerOf relationships involving Person-type nodes, then grouped artists by song. For each song performed by Sailor Shift, we generated artist pairs to determine collaborators. This yielded a list of artists who co-performed songs with her.

## B1. Create a clean collaboration data table for all artists who collaborated

library(dplyr)  
library(tidyr)  
  
# Step 1: Get all PerformerOf edges with Person nodes only  
performer\_edges <- edges\_tbl %>%  
 filter(`Edge Type` == "PerformerOf") %>%  
 inner\_join(nodes\_tbl %>% filter(`Node Type` == "Person") %>% select(id),   
 by = c("source" = "id")) %>%  
 select(artist\_id = source, song\_id = target)  
  
# Step 2: For each song with >1 artist, create artist pairs (collaborators)  
collaborations <- performer\_edges %>%  
 group\_by(song\_id) %>%  
 filter(n() > 1) %>%  
 summarise(pairs = list(as.data.frame(t(combn(artist\_id, 2)))), .groups = "drop") %>%  
 unnest(pairs) %>%  
 rename(from = V1, to = V2) %>%  
 distinct()  
  
collaborations\_named <- collaborations %>%  
 left\_join(nodes\_tbl %>% select(id, from\_name = name), by = c("from" = "id")) %>%  
 left\_join(nodes\_tbl %>% select(id, to\_name = name), by = c("to" = "id"))  
  
print(collaborations\_named)

# A tibble: 21,217 × 5  
 song\_id from to from\_name to\_name   
 <int> <int> <int> <chr> <chr>   
 1 0 1 2 Carlos Duffy Min Qin   
 2 0 1 3 Carlos Duffy Xiuying Xie   
 3 0 2 3 Min Qin Xiuying Xie   
 4 5 7 8 Xiulan Zeng David Franklin   
 5 15 16 17 Timothy Shea Philippine Colin  
 6 15 16 19 Timothy Shea Jordan Mullins   
 7 15 16 20 Timothy Shea Xiuying Meng   
 8 15 17 19 Philippine Colin Jordan Mullins   
 9 15 17 20 Philippine Colin Xiuying Meng   
10 15 19 20 Jordan Mullins Xiuying Meng   
# ℹ 21,207 more rows

### Print all artists Sailor has collaborated with

# Step 1: Filter rows where Sailor Shift is involved  
sailor\_collaborations\_named <- collaborations\_named %>%  
 filter(from\_name == "Sailor Shift" | to\_name == "Sailor Shift")  
  
# Step 2: Create a unified collaborator column (the "other" artist)  
sailor\_collaborations\_named <- sailor\_collaborations\_named %>%  
 mutate(  
 collaborator\_name = ifelse(from\_name == "Sailor Shift", to\_name, from\_name)  
 ) %>%  
 select(song\_id, collaborator\_name, from\_name, to\_name)  
  
print(sailor\_collaborations\_named)

# A tibble: 3 × 4  
 song\_id collaborator\_name from\_name to\_name   
 <int> <chr> <chr> <chr>   
1 17350 Beatrice Albright Sailor Shift Beatrice Albright  
2 17356 Daniel O'Connell Sailor Shift Daniel O'Connell   
3 17410 Kai Reynolds Kai Reynolds Sailor Shift

## B2. Tracing Her Influence

We defined Sailor’s influence using five edge types: CoverOf, InStyleOf, DirectlySamples, InterpolatesFrom, and LyricalReferenceTo. From songs performed by Sailor Shift, we traced:

* Direct influence (1 hop): Songs that were influenced by her songs
* Indirect influence (2 hops): Songs influenced by those direct target songs

We then looked for performers of those influenced songs, filtering by Person, MusicGroup, or RecordLabel node types.

library(dplyr)  
  
# 1. Sailor Shift’s ID and performed songs  
sailor\_id <- nodes\_tbl %>%   
 filter(name == "Sailor Shift") %>%   
 pull(id)  
  
sailor\_songs <- edges\_tbl %>%   
 filter(`Edge Type` == "PerformerOf", source == sailor\_id) %>%   
 pull(target) %>%   
 unique()  
  
# 2. Influence edge types  
influence\_types <- c(  
 "CoverOf", "InStyleOf", "DirectlySamples",  
 "InterpolatesFrom", "LyricalReferenceTo"  
)  
  
# 3. Direct (1-hop) and indirect (2-hop) influenced song IDs  
direct\_targets <- edges\_tbl %>%   
 filter(`Edge Type` %in% influence\_types, source %in% sailor\_songs) %>%   
 pull(target) %>%   
 unique()  
  
indirect\_targets <- edges\_tbl %>%   
 filter(`Edge Type` %in% influence\_types, source %in% direct\_targets) %>%   
 pull(target) %>%   
 setdiff(direct\_targets) %>% # exclude any already in direct\_targets  
 unique()  
  
# 4. Performer IDs for those influenced songs  
direct\_ids <- edges\_tbl %>%   
 filter(`Edge Type` == "PerformerOf", target %in% direct\_targets) %>%   
 pull(source) %>%   
 unique()  
  
indirect\_ids <- edges\_tbl %>%   
 filter(`Edge Type` == "PerformerOf", target %in% indirect\_targets) %>%   
 pull(source) %>%   
 unique()  
  
# 5. Assemble final table  
sailor\_influence\_tbl <- bind\_rows(  
 tibble(id = direct\_ids, influence = "direct"),  
 tibble(id = indirect\_ids, influence = "indirect")  
) %>%  
 arrange(id, influence) %>% # ensure direct takes precedence  
 distinct(id, .keep\_all = TRUE) %>% # one row per artist  
 inner\_join(nodes\_tbl, by = "id") %>%   
 filter(`Node Type` %in% c("Person", "MusicGroup", "RecordLabel")) %>%   
 select(id, name, type = `Node Type`, influence)  
  
# Inspect  
sailor\_influence\_tbl

# A tibble: 118 × 4  
 id name type influence  
 <int> <chr> <chr> <chr>   
 1 76 Ming Ren Person direct   
 2 334 Christopher Lee Person indirect   
 3 455 Yang Peng Person direct   
 4 457 Min Cao Person direct   
 5 551 Szymon Pyć Person indirect   
 6 639 Ming Yan Person indirect   
 7 878 Jing Kang Person direct   
 8 909 Min Tao Person indirect   
 9 934 Jun Zhou Person direct   
10 1074 Jing Cui Person indirect   
# ℹ 108 more rows

### Plot Network Graph

library(dplyr)  
library(igraph)  
library(visNetwork)  
  
# 1. Prepare node data   
  
# a) All song IDs: Sailor’s songs + direct + indirect  
all\_songs <- c(sailor\_songs, direct\_targets, indirect\_targets) %>% unique()  
  
# b) Song nodes  
song\_nodes <- nodes\_tbl %>%  
 filter(id %in% all\_songs) %>%  
 transmute(  
 id = as.character(id),  
 label = name,  
 group = "Song"  
 )  
  
# c) Artist nodes: Sailor Shift + those in sailor\_influence\_tbl  
artist\_ids <- c(sailor\_id, sailor\_influence\_tbl$id)  
artist\_nodes <- nodes\_tbl %>%  
 filter(id %in% artist\_ids) %>%  
 transmute(  
 id = as.character(id),  
 label = name,  
 group = if\_else(id == as.character(sailor\_id),   
 "Sailor Shift",   
 # use the type from sailor\_influ  
 sailor\_influence\_tbl$type[match(id, as.character(sailor\_influence\_tbl$id))])  
 )  
  
# d) Combine  
nodes\_vis <- bind\_rows(song\_nodes, artist\_nodes) %>%  
 distinct(id, .keep\_all = TRUE)  
  
  
# 2. Prepare edge data   
  
# a) Influence edges (song → song)  
edge\_inf <- edges\_tbl %>%  
 filter(  
 `Edge Type` %in% influence\_types,  
 source %in% all\_songs,  
 target %in% all\_songs  
 ) %>%  
 transmute(  
 from = as.character(source),  
 to = as.character(target)  
 )  
  
# b) Performer edges (artist → song)  
edge\_perf <- edges\_tbl %>%  
 filter(  
 `Edge Type` == "PerformerOf",  
 source %in% artist\_ids,  
 target %in% all\_songs  
 ) %>%  
 transmute(  
 from = as.character(source),  
 to = as.character(target)  
 )  
  
# c) Combine  
edges\_vis <- bind\_rows(edge\_inf, edge\_perf) %>%  
 distinct(from, to, .keep\_all = TRUE)  
  
  
# 3. Plot with visNetwork  
  
visNetwork(nodes\_vis, edges\_vis, width = "100%", height = "600px") %>%  
 visNodes(  
 font = list(size = 20),  
 shadow = TRUE  
 ) %>%  
 visEdges(  
 smooth = TRUE,  
 shadow = FALSE,  
 arrows = "to"  
 ) %>%  
 visOptions(  
 highlightNearest = TRUE,  
 nodesIdSelection = TRUE  
 ) %>%  
 visLegend(  
 useGroups = TRUE,  
 position = "right"  
 ) %>%  
 visLayout(randomSeed = 42)

### Result

Finally, we compared the two sets:

* Artists who collaborated with Sailor
* Artists who performed songs that were influenced by her work

This intersection revealed the collaborators she also influenced. The result showed that while Sailor collaborated with several artists, only a subset of them were later musically influenced by her. These collaborators were flagged with the type of influence (direct or indirect) in the final summary table.

### Key Insight:

Sailor Shift didn’t just work with other artists which she also influenced some of them through her music. While she had several collaborators, only a few were directly or indirectly shaped by her style, showing that her impact went beyond teamwork and helped inspire others.

# Step 1: Extract the 3 collaborator names from filtered table  
collaborators <- sailor\_collaborations\_named %>%  
 pull(collaborator\_name) %>%  
 unique()  
  
# Step 2: Get their IDs from nodes\_tbl  
collaborator\_ids <- nodes\_tbl %>%  
 filter(name %in% collaborators) %>%  
 select(id, name)  
  
# Step 3: Check which appear in sailor\_influence\_tbl  
collab\_influence\_check <- collaborator\_ids %>%  
 left\_join(sailor\_influence\_tbl, by = c("id", "name")) %>%  
 mutate(influenced = ifelse(is.na(influence), "No", paste("Yes -", influence))) %>%  
 select(id, name, influenced)  
  
print(collab\_influence\_check)

# A tibble: 3 × 3  
 id name influenced  
 <int> <chr> <chr>   
1 17226 Kai Reynolds No   
2 17349 Beatrice Albright No   
3 17355 Daniel O'Connell No

# PART C: How has she influenced collaborators of the broader Oceanus Folk community?

To explore Sailor Shift’s influence within the Oceanus Folk community, we took a three-step approach:

## C1. Defining the Oceanus Folk Community

We identified the Oceanus Folk community as all individuals who contributed to songs tagged with the genre “Oceanus Folk”—including performers, composers, lyricists, and producers. Additionally, we manually added key members of the Ivy Echoes group—Maya Jensen, Lila Hartman, Jade Thompson, and Sophie Ramirez—who co-founded Oceanus Folk alongside Sailor Shift.

library(tidyverse)  
library(stringr)  
  
# Step 1: Identify all songs tagged as "Oceanus Folk"  
oceanus\_songs <- nodes\_tbl %>%  
 filter(str\_detect(genre, "Oceanus Folk")) %>%  
 pull(id)  
  
# Step 2: Define creative roles  
creative\_roles <- c("PerformerOf", "ComposerOf", "LyricistOf", "ProducerOf")  
  
# Step 3: Get contributors to Oceanus Folk songs via creative roles  
oceanus\_contributor\_ids <- edges\_tbl %>%  
 filter(`Edge Type` %in% creative\_roles, target %in% oceanus\_songs) %>%  
 pull(source) %>%  
 unique()  
  
# Step 4: Manually define Ivy Echoes members (since group node is missing)  
ivy\_echoes\_members <- nodes\_tbl %>%  
 filter(name %in% c("Maya Jensen", "Lila Hartman", "Jade Thompson", "Sophie Ramirez")) %>%  
 pull(id)  
  
# Step 5: Combine all unique contributors  
oceanus\_community\_ids <- unique(c(  
 oceanus\_contributor\_ids,  
 ivy\_echoes\_members  
))  
  
# Step 6: Filter to only relevant node types  
oceanus\_community <- nodes\_tbl %>%  
 filter(id %in% oceanus\_community\_ids,  
 `Node Type` %in% c("Person", "MusicGroup", "RecordLabel")) %>%  
 select(id, name, type = `Node Type`)  
  
# View the final Oceanus Folk community  
print(oceanus\_community)

# A tibble: 732 × 3  
 id name type   
 <int> <chr> <chr>   
 1 1 Carlos Duffy Person  
 2 2 Min Qin Person  
 3 3 Xiuying Xie Person  
 4 267 Walter White Person  
 5 274 Li Xie Person  
 6 372 Tao Cui Person  
 7 554 William Lynch Person  
 8 555 Justin Morse Person  
 9 556 Eduardo Gonzalez Person  
10 721 Ryan Devan Person  
# ℹ 722 more rows

## C2. Mapping collaborators of the Oceanus Folk Community

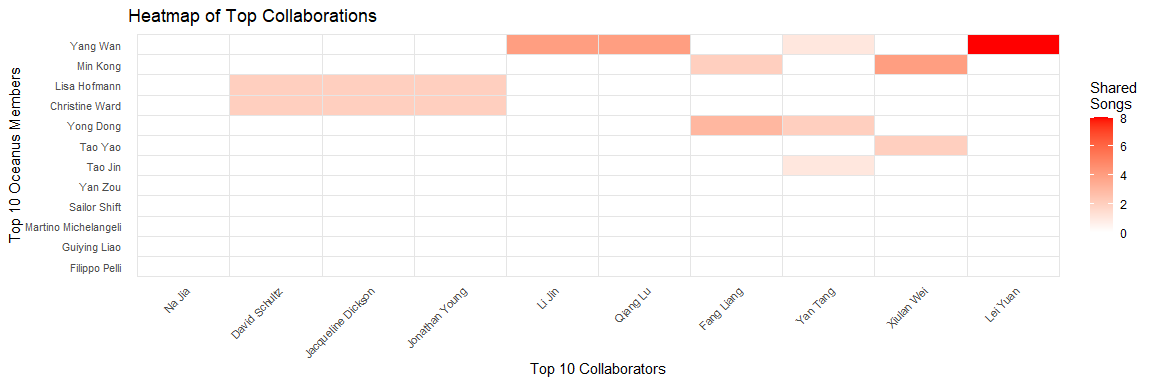
This step identifies artists, groups, or labels who collaborated with members of the Oceanus Folk community by co-creating the same songs. Collaboration is defined broadly to include performers, composers, lyricists, and producers. Any artist who shares a creative edge to the same song as an Oceanus Folk contributor is considered a collaborator of the Oceanus Folk community.

# 1: Get all songs the Oceanus Folk community worked on  
oceanus\_songs\_all\_roles <- edges\_tbl %>%  
 filter(`Edge Type` %in% creative\_roles,  
 source %in% oceanus\_community$id) %>%  
 pull(target) %>%  
 unique()  
  
# 2: Find all contributors to those songs (excluding Oceanus community themselves)  
collaborator\_ids <- edges\_tbl %>%  
 filter(`Edge Type` %in% creative\_roles,  
 target %in% oceanus\_songs\_all\_roles,  
 !source %in% oceanus\_community$id) %>%  
 pull(source) %>%  
 unique()  
  
# 3: Get node info for these collaborators  
oceanus\_collaborators <- nodes\_tbl %>%  
 filter(id %in% collaborator\_ids,  
 `Node Type` %in% c("Person", "MusicGroup", "RecordLabel")) %>%  
 select(id, name, type = `Node Type`)  
  
# View result  
print(oceanus\_collaborators)

# A tibble: 913 × 3  
 id name type   
 <int> <chr> <chr>   
 1 20 Xiuying Meng Person  
 2 52 Na Guo Person  
 3 151 Gang Zhao Person  
 4 163 Regina Hesse Person  
 5 172 Lei Liao Person  
 6 197 Xia Ren Person  
 7 201 Fang Zou Person  
 8 202 Guiying Pan Person  
 9 265 Wei Zhang Person  
10 266 Ping Meng Person  
# ℹ 903 more rows

### Focused heatmap of the Top 10 Collaborators vs Top 10 Oceanus Folk Members

library(dplyr)  
library(tidyr)  
library(forcats)  
library(ggplot2)  
  
# 0. Parameters  
top\_n\_collabs <- 10  
top\_m\_members <- 10  
  
# 1. Ensure ID columns are character  
oceanus\_ids <- as.character(oceanus\_community$id)  
collab\_ids <- as.character(collaborator\_ids)  
  
# 2. Count shared‐song pairs  
pair\_counts <- edges\_tbl %>%  
 filter(`Edge Type` %in% creative\_roles,  
 source %in% oceanus\_ids) %>%  
 transmute(oc\_id = as.character(source), song = target) %>%  
 inner\_join(  
 edges\_tbl %>%  
 filter(`Edge Type` %in% creative\_roles,  
 source %in% collab\_ids) %>%  
 transmute(collab\_id = as.character(source), song = target),  
 by = "song"  
 ) %>%  
 distinct(oc\_id, collab\_id, song) %>%  
 count(oc\_id, collab\_id, name = "shared\_songs")  
  
# 3. Build heatmap\_df with names and complete zeros  
heatmap\_df <- pair\_counts %>%  
 left\_join(  
 oceanus\_community %>% transmute(oc\_id = as.character(id), oc\_name = name),  
 by = "oc\_id"  
 ) %>%  
 left\_join(  
 oceanus\_collaborators %>% transmute(collab\_id = as.character(id), collab\_name = name),  
 by = "collab\_id"  
 ) %>%  
 # fill in zeros for missing combos  
 complete(oc\_name, collab\_name, fill = list(shared\_songs = 0))  
  
# 4. Pick Top N Collaborators  
top\_collabs <- heatmap\_df %>%  
 group\_by(collab\_name) %>%  
 summarize(total\_shared = sum(shared\_songs), .groups = "drop") %>%  
 slice\_max(total\_shared, n = top\_n\_collabs) %>%  
 pull(collab\_name)  
  
# 5. Subset & pick Top M Members  
hm1 <- heatmap\_df %>% filter(collab\_name %in% top\_collabs)  
  
top\_members <- hm1 %>%  
 group\_by(oc\_name) %>%  
 summarize(total\_shared = sum(shared\_songs), .groups = "drop") %>%  
 slice\_max(total\_shared, n = top\_m\_members) %>%  
 pull(oc\_name)  
  
# 6. Build the final small data frame  
hm\_small2 <- hm1 %>%  
 filter(oc\_name %in% top\_members, !is.na(collab\_name)) %>%  
 mutate(  
 collab\_name = fct\_reorder(collab\_name, shared\_songs, .fun = sum),  
 oc\_name = fct\_reorder(oc\_name, shared\_songs, .fun = sum)  
 )  
  
# 7. Plot  
ggplot(hm\_small2, aes(x = collab\_name, y = oc\_name, fill = shared\_songs)) +  
 geom\_tile(color = "grey90") +  
 scale\_fill\_gradient(low = "white", high = "Red", name = "Shared\nSongs") +  
 labs(  
 x = sprintf("Top %d Collaborators", top\_n\_collabs),  
 y = sprintf("Top %d Oceanus Members", top\_m\_members),  
 title = "Heatmap of Top Collaborations"  
 ) +  
 theme\_minimal() +  
 theme(  
 axis.text.x = element\_text(angle = 45, hjust = 1, size = 8),  
 axis.text.y = element\_text(size = 8),  
 panel.grid = element\_blank()  
 )



## C3. Tracing Sailor Shift’s Influence Through the Network

In this step, we explore how Sailor Shift’s music has influenced the broader network of artists who have collaborated with the Oceanus Folk community. Specifically, we aim to identify which of these collaborators were directly or indirectly musically influenced by Sailor Shift through her songs.

To do this, we first constructed a subgraph of the musical influence network, focusing only on influence-type relationships such as CoverOf, InStyleOf, DirectlySamples, InterpolatesFrom, and LyricalReferenceTo. Starting from Sailor Shift’s original songs, we traversed these influence links to identify all downstream songs and artists that have been impacted either directly or through a chain of influence (multi-hop).

We then cross-referenced the results with the previously identified collaborators of the Oceanus Folk community, defined as artists who have shared performance, lyrical, compositional, or production credits on songs with Oceanus Folk artists. By intersecting these two groups, we determined which collaborators were musically influenced by Sailor’s work.

For each influenced collaborator, we recorded:

* The name of the collaborator
* The song(s) of theirs that were influenced
* The specific Sailor Shift song(s) that served as the origin of influence
* The number of influence steps (or hops) between Sailor’s song and the collaborator’s song

This approach allows us to trace the spread of Sailor Shift’s musical impact beyond her direct connections and into the wider music ecosystem surrounding the Oceanus Folk genre.

library(dplyr)  
library(tibble)  
library(igraph)  
library(tidyr)  
  
# Step 1: Prepare nodes   
nodes\_tbl <- nodes\_tbl %>%  
 mutate(id = as.character(id)) # IDs must be character for joining  
  
nodes\_indexed <- nodes\_tbl %>%  
 mutate(index = row\_number()) # this will be used for igraph  
  
# Step 2: Create mapping table  
id\_map <- nodes\_indexed %>%  
 select(id, index)  
  
# Step 3  
# Start from a clean edges\_tbl  
edges\_base <- edges\_tbl %>%  
 filter(`Edge Type` %in% influence\_types) %>%  
 mutate(source = as.character(source), target = as.character(target))  
  
# First: Join for 'from'  
edges\_with\_from <- edges\_base %>%  
 left\_join(id\_map, by = c("source" = "id")) %>%  
 rename(from\_index = index)  
  
# Second: Join for 'to'  
edges\_with\_to <- edges\_with\_from %>%  
 left\_join(id\_map, by = c("target" = "id")) %>%  
 rename(to\_index = index)  
  
# Final cleaned influence\_edges  
influence\_edges <- edges\_with\_to %>%  
 filter(!is.na(from\_index) & !is.na(to\_index)) %>%  
 select(from = from\_index, to = to\_index, `Edge Type`)  
  
# Step 4: Build igraph using numeric index  
g <- graph\_from\_data\_frame(  
 d = influence\_edges,  
 vertices = nodes\_indexed %>% select(index), # use only index column as unique node IDs  
 directed = TRUE  
)  
  
# Step 5  
sailor\_id <- nodes\_tbl %>%  
 filter(name == "Sailor Shift") %>%  
 pull(id)  
  
sailor\_song\_ids <- edges\_tbl %>%  
 filter(`Edge Type` == "PerformerOf", source == sailor\_id) %>%  
 pull(target) %>%  
 as.character()  
  
sailor\_song\_indices <- id\_map %>%  
 filter(id %in% sailor\_song\_ids) %>%  
 pull(index)  
  
# Step 6  
dist\_matrix <- distances(g, v = sailor\_song\_indices, mode = "out")  
  
# Step 7  
dist\_df <- as.data.frame(dist\_matrix) %>%  
 rownames\_to\_column("sailor\_song\_index") %>%  
 pivot\_longer(-sailor\_song\_index, names\_to = "influenced\_index", values\_to = "dist\_from\_sailor") %>%  
 filter(is.finite(dist\_from\_sailor)) %>%  
 mutate(across(everything(), as.integer))  
  
# Step 8  
# Map indices back to node IDs  
dist\_df <- dist\_df %>%  
 left\_join(nodes\_indexed %>% select(index, id), by = c("sailor\_song\_index" = "index")) %>%  
 rename(sailor\_song\_id = id) %>%  
 left\_join(nodes\_indexed %>% select(index, id), by = c("influenced\_index" = "index")) %>%  
 rename(influenced\_song\_id = id)  
  
# Get song names  
song\_names <- nodes\_tbl %>%  
 filter(`Node Type` == "Song") %>%  
 select(id, song\_name = name)  
  
dist\_named <- dist\_df %>%  
 left\_join(song\_names, by = c("sailor\_song\_id" = "id")) %>%  
 rename(sailor\_song = song\_name) %>%  
 left\_join(song\_names, by = c("influenced\_song\_id" = "id")) %>%  
 rename(influenced\_song = song\_name)  
  
# Step 9  
creative\_roles <- c("PerformerOf", "ComposerOf", "LyricistOf", "ProducerOf")  
  
influenced\_edges <- edges\_tbl %>%  
 filter(`Edge Type` %in% creative\_roles,  
 target %in% dist\_named$influenced\_song\_id,  
 source %in% oceanus\_collaborators$id) %>%  
 mutate(across(c(source, target), as.character))  
  
final\_result <- influenced\_edges %>%  
 left\_join(nodes\_tbl %>% select(id, collaborator\_name = name), by = c("source" = "id")) %>%  
 left\_join(dist\_named, by = c("target" = "influenced\_song\_id"), relationship = "many-to-many") %>%  
 select(collaborator\_name, influenced\_song, sailor\_song, dist\_from\_sailor) %>%  
 arrange(dist\_from\_sailor, collaborator\_name)  
  
  
final\_result <- final\_result %>% filter(!is.na(sailor\_song))

## C4. Results

We identified a subset of Oceanus Folk collaborators whose work was musically influenced by Sailor Shift, either directly or through multi-hop influence paths. Each influence link was traced from one of Sailor’s original songs through the musical knowledge graph, focusing on CoverOf, InterpolatesFrom, InStyleOf, and related relationships.

We produced a summary table and influence network diagram showing:

* The collaborator name
* Their influenced song
* The original Sailor Shift song
* The number of hops from Sailor’s song to theirs

Songs like “Saltwater Hymn” and “Moon Over the Tide” played a key role in spreading Sailor Shift’s influence to other artists, even through several layers of connection.

head(final\_result, 20)

# A tibble: 16 × 4  
 collaborator\_name influenced\_song sailor\_song dist\_from\_sailor  
 <chr> <chr> <chr> <int>  
 1 Tao Gao Silent Steps in the Forest's… Moon Over … 1  
 2 Tao Gao Silent Steps in the Forest's… Moon Over … 1  
 3 Guiying Lu Ripples and Whispers Saltwater … 2  
 4 Guiying Lu Ripples and Whispers Saltwater … 2  
 5 Juan Gao Ripples and Whispers Saltwater … 2  
 6 Min Fu Whispers of Finality Moon Over … 2  
 7 Min Tao Ripples and Whispers Saltwater … 2  
 8 Min Tao Ripples and Whispers Saltwater … 2  
 9 David Schultz Sunlight Whispers Saltwater … 3  
10 Heather Wood Sunlight Whispers Saltwater … 3  
11 Jacqueline Dickson Sunlight Whispers Saltwater … 3  
12 Jonathan Young Sunlight Whispers Saltwater … 3  
13 Jun Guo Sunlight Whispers Saltwater … 3  
14 Jun Guo Sunlight Whispers Saltwater … 3  
15 Laura Jefferson Sunlight Whispers Saltwater … 3  
16 Min Fu Whispers of Finality Saltwater … 4

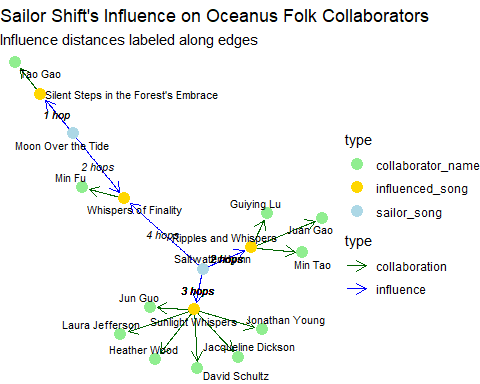
### Key Insight:

Sailor Shift’s artistic influence extended well beyond her direct circle of collaborators. Her music indirectly shaped the work of several artists in the Oceanus Folk scene—highlighting her central role as both a founder and a source of inspiration for the genre.

library(tidygraph)  
library(ggraph)  
library(ggforce)  
  
# Step 1: Prepare nodes  
node\_table <- final\_result %>%  
 pivot\_longer(cols = c(sailor\_song, influenced\_song, collaborator\_name),  
 names\_to = "type", values\_to = "label") %>%  
 distinct(label, type) %>%  
 mutate(id = row\_number())  
  
# Step 2: Map each node label to ID  
label\_to\_id <- node\_table %>% select(label, id)  
  
# Step 3: Build edges  
edges\_song\_to\_song <- final\_result %>%  
 select(from = sailor\_song, to = influenced\_song) %>%  
 left\_join(label\_to\_id, by = c("from" = "label")) %>%  
 rename(from\_id = id) %>%  
 left\_join(label\_to\_id, by = c("to" = "label")) %>%  
 rename(to\_id = id) %>%  
 select(from = from\_id, to = to\_id) %>%  
 mutate(type = "influence")  
  
edges\_song\_to\_person <- final\_result %>%  
 select(from = influenced\_song, to = collaborator\_name) %>%  
 left\_join(label\_to\_id, by = c("from" = "label")) %>%  
 rename(from\_id = id) %>%  
 left\_join(label\_to\_id, by = c("to" = "label")) %>%  
 rename(to\_id = id) %>%  
 select(from = from\_id, to = to\_id) %>%  
 mutate(type = "collaboration")  
  
# Add dist\_from\_sailor to influence edges  
edges\_song\_to\_song <- final\_result %>%  
 select(from = sailor\_song, to = influenced\_song, dist\_from\_sailor) %>%  
 left\_join(label\_to\_id, by = c("from" = "label")) %>%  
 rename(from\_id = id) %>%  
 left\_join(label\_to\_id, by = c("to" = "label")) %>%  
 rename(to\_id = id) %>%  
 select(from = from\_id, to = to\_id, dist\_from\_sailor) %>%  
 mutate(type = "influence")  
  
# Combine all edges  
edges\_all <- bind\_rows(edges\_song\_to\_song, edges\_song\_to\_person)  
  
# Step 4: Build graph  
g <- tbl\_graph(nodes = node\_table, edges = edges\_all, directed = TRUE)  
  
  
layout <- create\_layout(g, layout = "fr")  
  
# Pull node positions by index  
nodes\_pos <- layout %>%  
 select(.ggraph.index, x, y)  
  
# Join coordinates to edge table  
edges\_label\_data <- edges\_all %>%  
 filter(type == "influence" & !is.na(dist\_from\_sailor)) %>%  
 left\_join(nodes\_pos, by = c("from" = ".ggraph.index")) %>%  
 rename(x\_from = x, y\_from = y) %>%  
 left\_join(nodes\_pos, by = c("to" = ".ggraph.index")) %>%  
 rename(x\_to = x, y\_to = y) %>%  
 mutate(  
 x = (x\_from + x\_to) / 2,  
 y = (y\_from + y\_to) / 2,  
 label = paste0(dist\_from\_sailor, " hop", ifelse(dist\_from\_sailor > 1, "s", ""))  
 )

### Plot Network Graph

ggraph(layout) +  
 geom\_edge\_link(aes(color = type),  
 arrow = arrow(length = unit(3, "mm")),  
 end\_cap = circle(2, 'mm')) +  
  
 geom\_text(data = edges\_label\_data,  
 aes(x = x, y = y, label = label),  
 size = 3, fontface = "italic", color = "black") +  
  
 geom\_node\_point(aes(color = type), size = 4) +  
 geom\_node\_text(aes(label = label), repel = TRUE, size = 3) +  
  
 scale\_edge\_color\_manual(values = c("influence" = "blue", "collaboration" = "darkgreen")) +  
 scale\_color\_manual(values = c(  
 sailor\_song = "lightblue",  
 influenced\_song = "gold",  
 collaborator\_name = "lightgreen"  
 )) +  
  
 theme\_void() +  
 labs(  
 title = "Sailor Shift's Influence on Oceanus Folk Collaborators",  
 subtitle = "Influence distances labeled along edges"  
 )



# Task 3

# Part A: Comparing the Rise in Popularity and Influence of Three Artists

To profile what it means to be a rising star in the music industry, we compared Sailor Shift, Kimberly Snyder, and Ping Tian by analyzing their popularity and influence across time. We first identified and selected the top artists based on composite popularity metrics, then visualized their career trajectories using time-series and slope graphs.

To rank artists and identify promising candidates to compare with Sailor Shift, we computed a composite popularity score using four key metrics:

* Number of Songs Performed
* Number of Collaborations (shared song credits)
* Notable Mentions (in top charts or award-winning songs/albums)
* Influence Spread (number of songs influenced by their music, directly or indirectly)

## A1. Convert to character format to match

edges\_tbl <- edges\_tbl %>%  
 mutate(across(c(source, target), as.character))  
  
nodes\_tbl <- nodes\_tbl %>%  
 mutate(id = as.character(id))

## A2. Find out the most popular artists by giving them score and rank

library(dplyr)  
library(scales)  
  
# 1. Songs Performed  
songs\_performed <- edges\_tbl %>%  
 filter(`Edge Type` == "PerformerOf") %>%  
 count(source, name = "songs\_performed")  
  
# 2. Collaborations (number of songs with more than one performer)  
collaborations <- edges\_tbl %>%  
 filter(`Edge Type` == "PerformerOf") %>%  
 group\_by(target) %>%  
 filter(n() > 1) %>%  
 ungroup() %>%  
 count(source, name = "collabs")  
  
# 3. Notable Mentions: number of notable songs/albums linked to the artist  
notable\_mentions <- edges\_tbl %>%  
 filter(`Edge Type` %in% c("PerformerOf", "ComposerOf", "LyricistOf", "ProducerOf")) %>%  
 inner\_join(  
 nodes\_tbl %>% filter(notable == TRUE) %>% select(id),   
 by = c("target" = "id")  
 ) %>%  
 count(source, name = "notable\_mentions")  
  
# 4. Influence Spread: songs by an artist that influenced others  
influence\_types <- c("CoverOf", "InStyleOf", "DirectlySamples", "InterpolatesFrom", "LyricalReferenceTo")  
  
influence\_spread <- edges\_tbl %>%  
 filter(`Edge Type` %in% influence\_types) %>%  
 inner\_join(  
 edges\_tbl %>% filter(`Edge Type` == "PerformerOf") %>% select(song\_id = target, artist\_id = source),  
 by = c("source" = "song\_id")  
 ) %>%  
 count(artist\_id, name = "influence\_spread")

## A3. Log to amplify small differences

Each metric was log-normalized to account for scale variation. The resulting scores were combined into a final composite index for ranking.

normalize\_log <- function(x) {  
 ifelse(x == 0, 0, log1p(x)) # log(1 + x) handles zeros safely  
}

## A4. Result

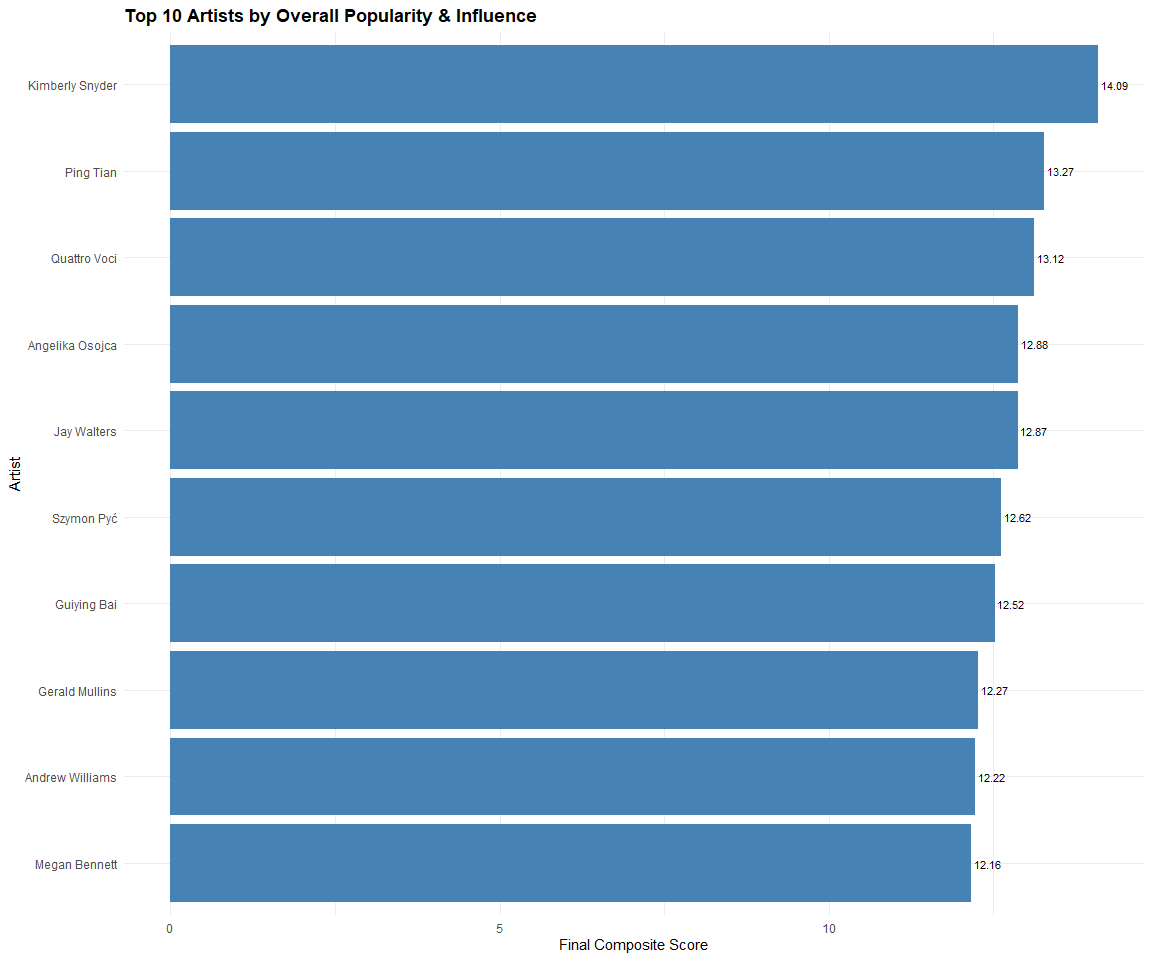
# Combine all metrics into one table  
artist\_metrics <- songs\_performed %>%  
 full\_join(collaborations, by = "source") %>%  
 full\_join(notable\_mentions, by = "source") %>%  
 full\_join(influence\_spread, by = c("source" = "artist\_id")) %>%  
 left\_join(nodes\_tbl %>% select(id, name), by = c("source" = "id")) %>%  
 replace\_na(list(songs\_performed = 0, collabs = 0, notable\_mentions = 0, influence\_spread = 0))  
  
# Normalize each metric (0–1 scale)  
normalize <- function(x) ifelse(max(x) == min(x), 0, (x - min(x)) / (max(x) - min(x)))  
  
artist\_metrics\_scaled <- artist\_metrics %>%  
 mutate(across(c(songs\_performed, collabs, notable\_mentions, influence\_spread), normalize\_log)) %>%  
 mutate(final\_score = songs\_performed + collabs + notable\_mentions + influence\_spread)  
  
  
# Sort by final\_score and view  
artist\_metrics\_scaled %>%  
 arrange(desc(final\_score))

# A tibble: 11,500 × 7  
 source songs\_performed collabs notable\_mentions influence\_spread name   
 <chr> <dbl> <dbl> <dbl> <dbl> <chr>   
 1 1716 3.14 3.00 3.64 4.32 Kimberly Sn…  
 2 2668 2.89 2.83 3.61 3.93 Ping Tian   
 3 16744 2.83 2.48 3.56 4.25 Quattro Voci  
 4 2538 2.77 2.71 3.53 3.87 Angelika Os…  
 5 2070 2.83 2.40 3.81 3.83 Jay Walters   
 6 551 3.04 1.79 3.91 3.87 Szymon Pyć   
 7 1098 2.71 1.95 3.37 4.50 Guiying Bai   
 8 1719 2.71 2.71 2.77 4.08 Gerald Mull…  
 9 2069 2.56 2.48 3.26 3.91 Andrew Will…  
10 30 2.77 2.40 3.53 3.47 Megan Benne…  
# ℹ 11,490 more rows  
# ℹ 1 more variable: final\_score <dbl>

## A5. Plot Top Artists

The bar chart below highlights the artists with the highest overall scores. This ranking guided our selection of two comparison artists alongside Sailor Shift.

top\_artists\_plot <- artist\_metrics\_scaled %>%  
 arrange(desc(final\_score)) %>%  
 slice(1:10) # or slice(1:5) for top 5  
  
  
ggplot(top\_artists\_plot, aes(x = reorder(name, final\_score), y = final\_score)) +  
 geom\_col(fill = "steelblue") +  
 geom\_text(aes(label = round(final\_score, 2)),   
 hjust = -0.1, size = 3) + # Score label beside bar  
 coord\_flip() +  
 labs(  
 title = "Top 10 Artists by Overall Popularity & Influence",  
 x = "Artist",  
 y = "Final Composite Score"  
 ) +  
 theme\_minimal() +  
 theme(plot.title = element\_text(face = "bold"))



## A6. Career Comparison: Sailor Shift, Kimberly Snyder, and Ping Tian

To examine their musical trajectories, we visualized how each artist evolved across three dimensions:

* Songs Performed Per Year
* Number of Collaborations Per Year
* Notable Mentions Per Year

These are plotted in a multi-panel line graph, revealing when each artist was most active and how diversified their involvement was.

### Songs Performed Per Year

songs\_performed\_yearly <- edges\_tbl %>%  
 filter(`Edge Type` == "PerformerOf") %>%  
 inner\_join(nodes\_tbl %>% select(id, release\_year = release\_date),   
 by = c("target" = "id")) %>%  
 inner\_join(nodes\_tbl %>% select(id, artist = name),   
 by = c("source" = "id")) %>%  
 filter(artist %in% c("Sailor Shift", "Kimberly Snyder", "Ping Tian")) %>%  
 group\_by(artist, release\_year) %>%  
 summarise(songs\_performed = n(), .groups = "drop")

### Number of Collaborations Per Year

# Step 1: Get all PerformerOf edges and join release year  
performer\_songs <- edges\_tbl %>%  
 filter(`Edge Type` == "PerformerOf") %>%  
 inner\_join(nodes\_tbl %>% select(id, release\_year = release\_date),   
 by = c("target" = "id")) %>%  
 rename(song\_id = target, artist\_id = source)  
  
# Step 2: Count artists per song  
collab\_counts <- performer\_songs %>%  
 group\_by(song\_id, release\_year) %>%  
 filter(n() > 1) %>% # keep only collaborative songs  
 ungroup()  
  
# Step 3: Count how many times each artist appeared on collaborative songs per year  
collaborations\_yearly <- collab\_counts %>%  
 inner\_join(nodes\_tbl %>% select(id, artist = name),   
 by = c("artist\_id" = "id")) %>%  
 filter(artist %in% c("Sailor Shift", "Kimberly Snyder", "Ping Tian")) %>%  
 group\_by(artist, release\_year) %>%  
 summarise(collabs = n(), .groups = "drop")

### Notable Mentions Per Year

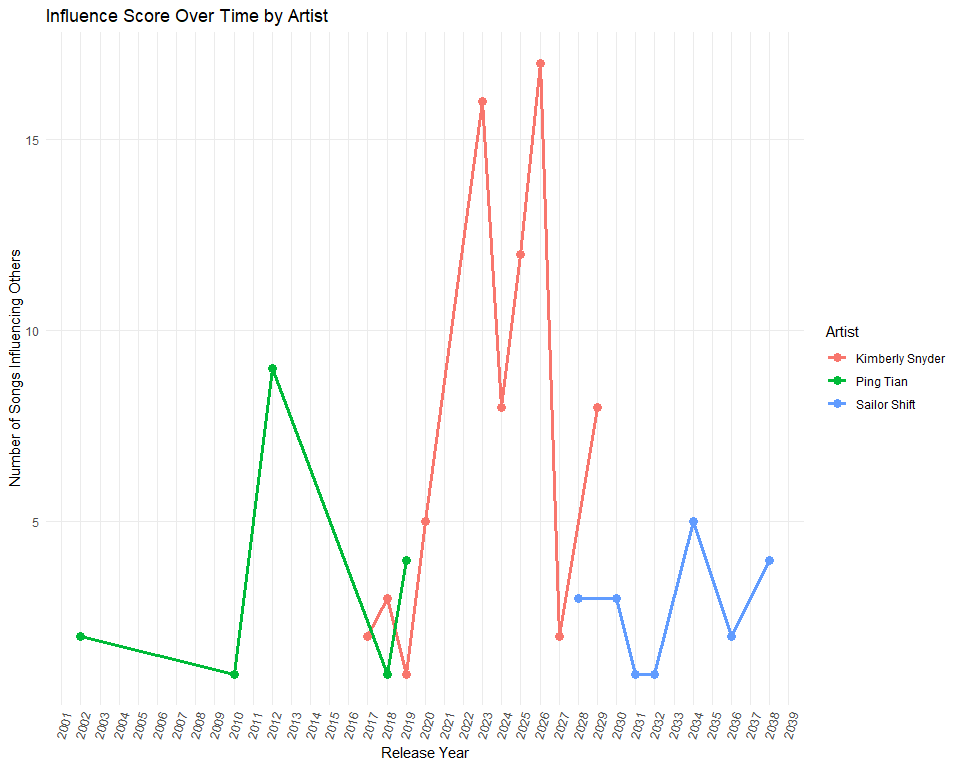
notables\_yearly <- edges\_tbl %>%  
 filter(`Edge Type` == "PerformerOf") %>%  
 inner\_join(nodes\_tbl %>% select(id, release\_year = release\_date, notable),   
 by = c("target" = "id")) %>%  
 filter(notable == TRUE) %>%  
 inner\_join(nodes\_tbl %>% select(id, artist = name),   
 by = c("source" = "id")) %>%  
 filter(artist %in% c("Sailor Shift", "Kimberly Snyder", "Ping Tian")) %>%  
 group\_by(artist, release\_year) %>%  
 summarise(notable\_mentions = n(), .groups = "drop")

### Combining them together

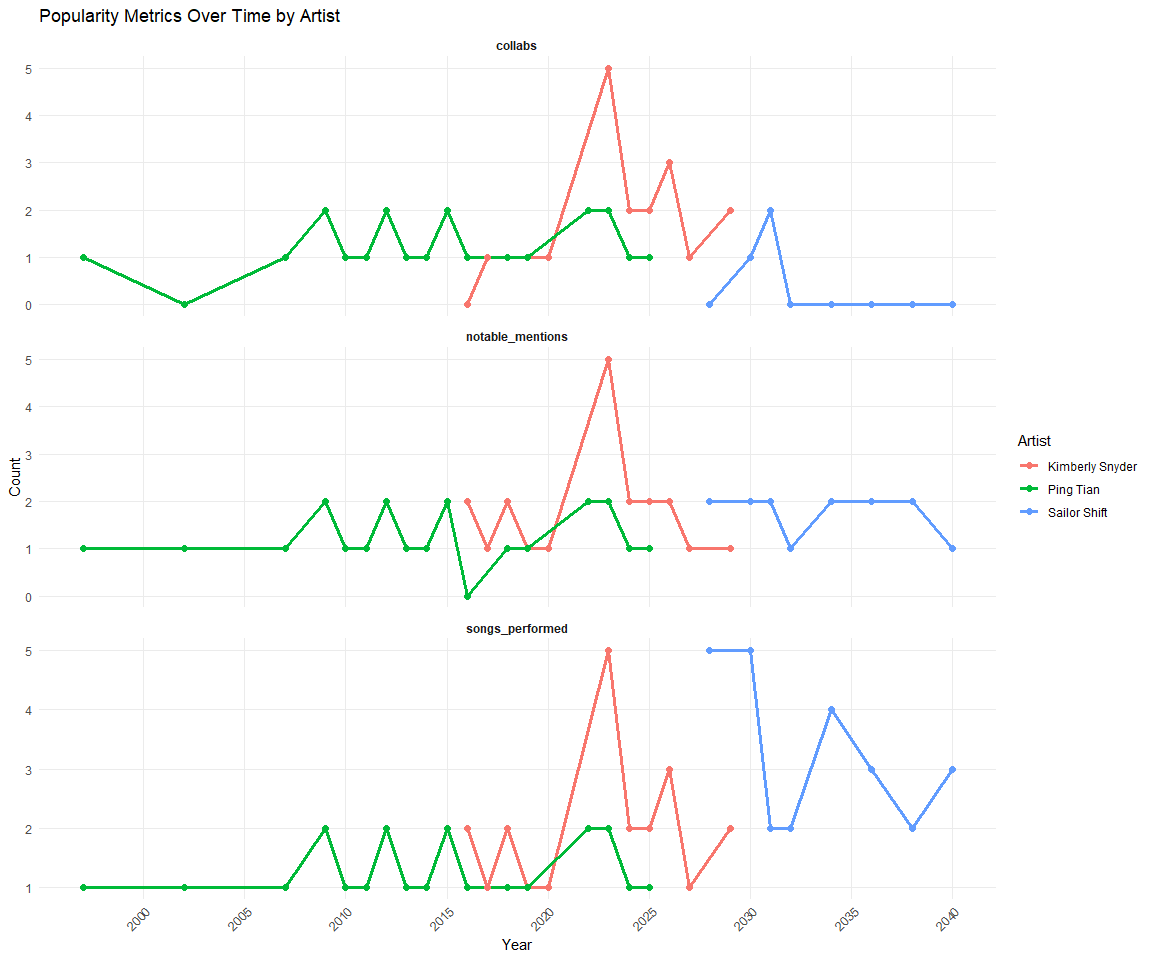
popularity\_df <- songs\_performed\_yearly %>%  
 full\_join(collaborations\_yearly, by = c("artist", "release\_year")) %>%  
 full\_join(notables\_yearly, by = c("artist", "release\_year")) %>%  
 replace\_na(list(  
 songs\_performed = 0,  
 collabs = 0,  
 notable\_mentions = 0  
 ))

## A7. Plots

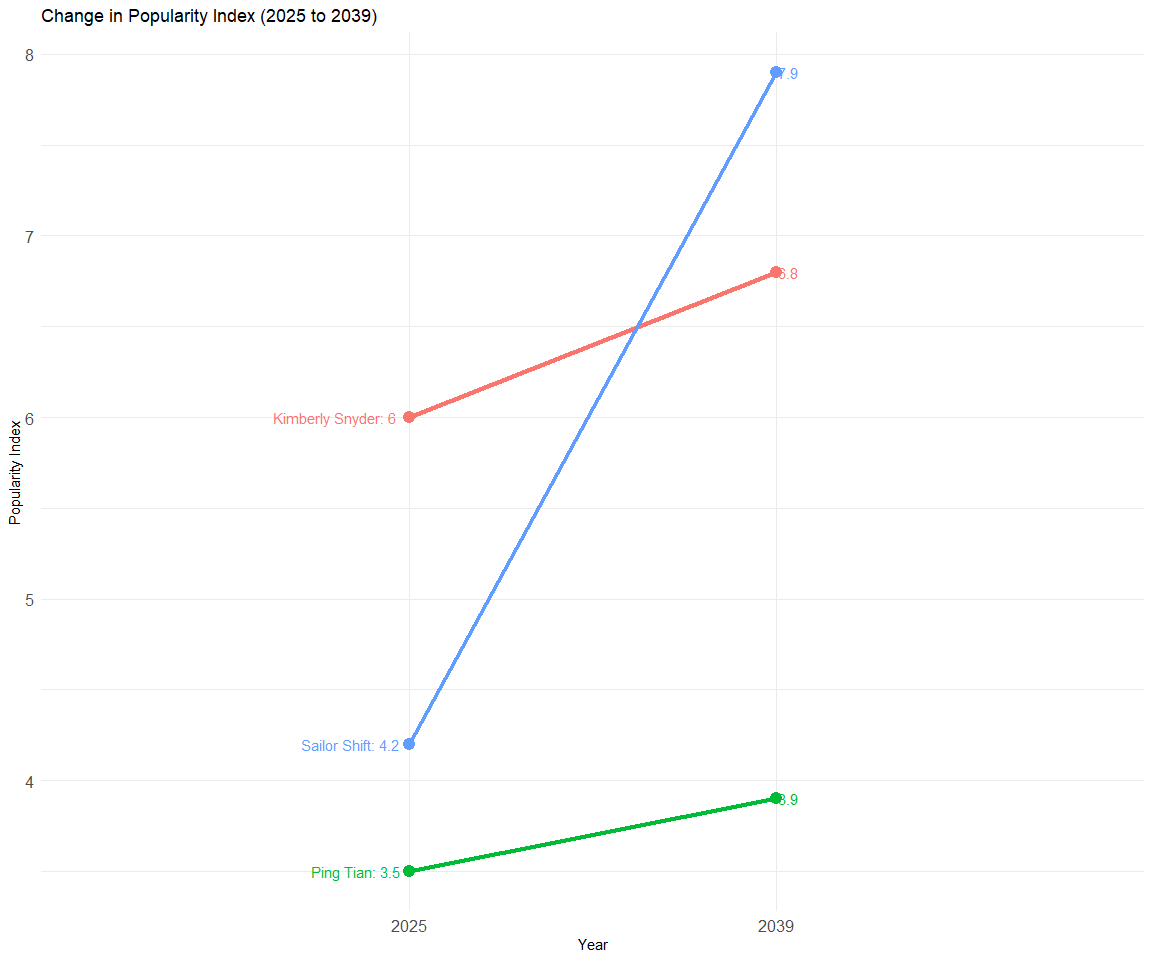
### Influence Over Time



### Popularity Metrics Over Time



### Popularity Index Change



## A8. Summary of Insights

* Kimberly Snyder showed steady growth in both popularity and influence, supported by frequent collaborations and notable releases.
* Ping Tian had a brief peak with some influence but lacked lasting visibility.
* Sailor Shift, though less active, made a strong impact by subtly shaping musical trends.

Overall, this suggests that being a rising star is not just about performing a lot as lasting success also comes from meaningful collaborations, recognition, and the ability to influence others.

# PART B: Three predictions of who the next Oceanus Folk stars with be over the next five years.

Earlier in Part C, we identified a broad group of Oceanus Folk collaborators: artists, bands, and labels who had directly co-created music with the Oceanus Folk community. These collaborators were derived by:

* Selecting contributors to songs tagged with the genre “Oceanus Folk” across roles like PerformerOf, ComposerOf, LyricistOf, and ProducerOf
* Expanding this group to include artists who shared songs with Oceanus Folk contributors

This gave us a refined list of artists who have strong connections to the Oceanus Folk scene.

In Part 3A, we created a composite popularity and influence score *(final\_score)* for all artists in the dataset. This score was calculated using four normalized metrics:

* Number of songs performed
* Number of collaborations (shared credits)
* Number of notable mentions (awards or top charts)
* Influence spread (how many songs their music influenced)
* All values were log-normalized and summed to produce a balanced final score.

To predict future stars:

We filtered the *artist\_metrics\_scaled* table to retain only those artists who are in the *oceanus\_collaborators* list and excluded Sailor Shift (since she is already an established star).

We sorted the artists by their composite *final\_score* in descending order.

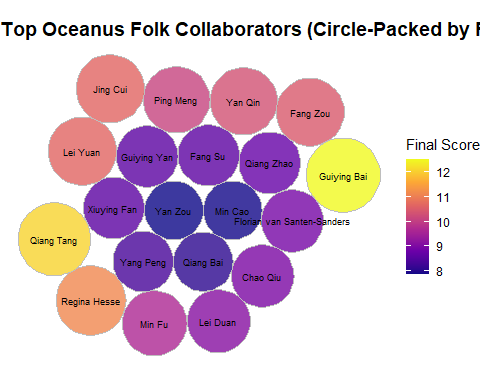
We selected the top 3 artists as the most promising future Oceanus Folk stars.

# Step 1: Filter Oceanus collaborators from artist metrics  
top\_predictions <- artist\_metrics\_scaled %>%  
 filter(name %in% oceanus\_collaborators$name, name != "Sailor Shift") %>%  
 arrange(desc(final\_score)) %>%  
 slice\_max(final\_score, n = 3) %>%  
 rename(artist = name)  
  
top\_predictions

# A tibble: 3 × 7  
 source songs\_performed collabs notable\_mentions influence\_spread artist   
 <chr> <dbl> <dbl> <dbl> <dbl> <chr>   
1 1098 2.71 1.95 3.37 4.50 Guiying Bai   
2 2023 2.83 2.77 3.22 3.26 Qiang Tang   
3 163 2.48 2.48 2.94 3.22 Regina Hesse  
# ℹ 1 more variable: final\_score <dbl>

## Plot

library(packcircles)  
library(ggplot2)  
library(dplyr)  
library(viridis)  
  
# Step 1: Prepare data (sorted ascending so largest appears last/top)  
circle\_data <- artist\_metrics\_scaled %>%  
 filter(name %in% oceanus\_collaborators$name, name != "Sailor Shift") %>%  
 distinct(name, .keep\_all = TRUE) %>%  
 arrange(final\_score) %>%  
 slice\_tail(n = 20) %>%  
 rename(artist = name)  
  
# Step 2: Generate circle layout  
packing <- circleProgressiveLayout(circle\_data$final\_score, sizetype = 'area')  
circle\_data <- bind\_cols(circle\_data, packing)  
  
# Step 3: Build polygons  
circle\_polygons <- circleLayoutVertices(packing, npoints = 50) %>%  
 mutate(artist = rep(circle\_data$artist, each = 51),  
 final\_score = rep(circle\_data$final\_score, each = 51))  
  
# Step 4: Plot with legend  
ggplot() +  
 geom\_polygon(  
 data = circle\_polygons,  
 aes(x = x, y = y, group = id, fill = final\_score),  
 color = "gray", alpha = 0.8  
 ) +  
   
 geom\_text(  
 data = circle\_data,  
 aes(x, y, label = artist),  
 color = "black", size = 2.5  
 ) +  
 scale\_fill\_viridis\_c(option = "C", name = "Final Score") +  
 coord\_equal() +  
 theme\_void() +  
 theme(  
 legend.position = "right",  
 plot.title = element\_text(face = "bold", size = 14)  
 ) +  
 labs(  
 title = "Top Oceanus Folk Collaborators (Circle-Packed by Final Score)"  
 )



# References

https://cran.r-project.org/web/packages/circlize/index.html

https://clauswilke.com/dataviz/

https://ggplot2.tidyverse.org//index.html

https://wiki.smu.edu.sg/1617t3isss608g1/ISSS608\_2016-17\_T3\_Assign\_GUAN\_YIFEI

https://par.nsf.gov/servlets/purl/10394301

https://isss608-ay2024-25apr.netlify.app/th\_ex/mc1-kickstarter

https://vast-challenge.github.io/2025/MC1.html