

Package: fairGNN (via r-universe)

June 8, 2026

Title Fairness-Aware Gated Neural Networks

Version 0.1.0

Description Tools for training and analysing fairness-aware gated neural networks for subgroup-aware prediction and interpretation in clinical datasets. Methods draw on prior work in mixture-of-experts neural networks by Jordan and Jacobs (1994) <[doi:10.1007/978-1-4471-2097-1_113](https://doi.org/10.1007/978-1-4471-2097-1_113)>, fairness-aware learning by Hardt, Price, and Srebro (2016) <[doi:10.48550/arXiv.1610.02413](https://doi.org/10.48550/arXiv.1610.02413)>, and personalised treatment prediction for depression by Iniesta, Stahl, and McGuffin (2016) <[doi:10.1016/j.jpsychires.2016.03.016](https://doi.org/10.1016/j.jpsychires.2016.03.016)>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports dplyr, tibble, ggplot2, readr, pROC, magrittr, tidyr, purrr, utils, stats, ggalluvial, tidyselect

Suggests knitr, torch, testthat, readxl, rmarkdown

VignetteBuilder knitr

URL <https://github.com/rhysholland/fairGNN>

BugReports <https://github.com/rhysholland/fairGNN/issues>

Depends R (>= 4.1.0)

SystemRequirements Optional 'LibTorch' backend; install via torch::install_torch().

NeedsCompilation no

Author Rhys Holland [aut, cre]

Maintainer Rhys Holland <rhys.holland@icloud.com>

Config/pak/sysreqs libicu-dev libx11-dev

Repository <https://rhysholland.r-universe.dev>

Date/Publication 2025-10-26 20:33:49 UTC

RemoteUrl <https://github.com/cran/fairGNN>

RemoteRef HEAD

RemoteSha c9ca1a118f20fea0af045321209166deed61d6f1

Contents

analyse_experts	2
analyse_gnn_results	3
analyse_gnn_results_plus	4
plot_sankey	4
prepare_data	5
train_gnn	6

Index **8**

analyse_experts	<i>Analyse and Visualise Expert Network Specialisation</i>
-----------------	--

Description

This function analyses the input weights of the expert sub-networks to determine which features are most important for each subgroup. It supports datasets with two or more subgroups.

Usage

```
analyse_experts(  
  gnn_results,  
  prepared_data,  
  group_mappings,  
  top_n_features = 10,  
  verbose = FALSE  
)
```

Arguments

gnn_results	A list object from train_gnn().
prepared_data	A list object from prepare_data().
group_mappings	A named list mapping numeric codes to labels.
top_n_features	The number of top features to visualise.
verbose	Logical, whether to print progress messages (default FALSE).

Value

A list containing the importance tables and plots.

analyse_gnn_results *Analyse and Visualize GNN Results*

Description

Generates plots and statistical tests for interpreting the GNN model results.

Usage

```
analyse_gnn_results(  
  gnn_results,  
  prepared_data,  
  group_mappings,  
  create_roc_plot = TRUE,  
  create_calibration_plot = TRUE,  
  analyse_gate_weights = TRUE,  
  analyse_gate_entropy = TRUE,  
  verbose = FALSE  
)
```

Arguments

gnn_results A list object from the `train_gnn()` function.

prepared_data A list object from the `prepare_data()` function.

group_mappings A named list that maps the numeric group codes back to their character labels for plotting (e.g., `list('0' = "Male", '1' = "Female")`).

create_roc_plot Boolean, if TRUE, generates and returns a ROC curve plot.

create_calibration_plot Boolean, if TRUE, generates and returns a calibration plot.

analyse_gate_weights Boolean, if TRUE, performs gate weight analysis (density plot and t-test).

analyse_gate_entropy Boolean, if TRUE, performs gate entropy analysis.

verbose Logical, whether to print progress messages (default FALSE).

Value

A list containing ggplot objects and analysis tables.

analyse_gnn_results_plus
Analyse GNN Results (+ Gate Summary Tables)

Description

Wraps analyse_gnn_results() and augments the return object with gate_weight_summary and gate_entropy_summary computed from gnn_results\$gate_weights.

Usage

```
analyse_gnn_results_plus(gnn_results, prepared_data, group_mappings, ...)
```

Arguments

gnn_results A list from train_gnn() (must include \$gate_weights).
 prepared_data A list from prepare_data().
 group_mappings A named vector/list mapping group codes to labels.
 ... Additional arguments passed through to analyse_gnn_results().

Value

The list returned by analyse_gnn_results() with two extra elements: gate_weight_summary and gate_entropy_summary.

plot_sankey *Create a Sankey Plot to Visualise Patient Routing*

Description

Create a Sankey Plot to Visualise Patient Routing

Usage

```
plot_sankey(  
  raw_data,  
  gnn_results,  
  expert_results,  
  group_mappings,  
  group_var,  
  verbose = FALSE  
)
```

Arguments

raw_data	The original, unscaled dataframe (must contain the features referenced by expert analysis).
gnn_results	The results object from <code>train_gnn()</code> (uses <code>\$final_results</code> and <code>\$gate_weights</code>).
expert_results	The results object from <code>analyse_experts()</code> .
group_mappings	A named list or named character vector mapping <i>codes</i> to <i>labels</i> (e.g., <code>c("0"="Male", "1"="Female")</code>). If provided in the reverse orientation (<i>labels</i> → <i>codes</i>), or unnamed, this function will normalise it automatically.
group_var	A string with the column name of the sensitive attribute in the <code>raw_data</code> .
verbose	Logical, whether to print progress messages (default <code>FALSE</code>).

Value

A ggplot object representing the Sankey diagram.

prepare_data	<i>Prepare Data for GNN Training</i>
--------------	--------------------------------------

Description

This function takes a raw dataframe, cleans it, defines the outcome and group variables, and scales the feature matrix.

Usage

```
prepare_data(
  data,
  outcome_var,
  group_var,
  group_mappings,
  cols_to_remove = NULL
)
```

Arguments

data	A dataframe containing the raw data.
outcome_var	A string with the column name of the binary outcome (must be 0 or 1).
group_var	A string with the column name of the sensitive attribute.
group_mappings	A named list that maps the values in <code>group_var</code> to numeric codes (0, 1, 2...). For example, <code>list("Male" = 0, "Female" = 1)</code> .
cols_to_remove	A character vector of column names to exclude from the feature matrix (e.g., IDs, highly collinear vars).

Value

A list containing:

X	The scaled feature matrix.
y	The numeric outcome vector.
group	The numeric group vector.
feature_names	The names of the features used.
subject_ids	A vector of subject IDs, if a 'subjectid' column exists.

Examples

```
# Fictional data example
my_data <- data.frame(
  subjectid = 1:10,
  remission = sample(0:1, 10, replace = TRUE),
  gender = sample(c("M", "F"), 10, replace = TRUE),
  feature1 = rnorm(10),
  feature2 = rnorm(10)
)

prepared_data <- prepare_data(
  data = my_data,
  outcome_var = "remission",
  group_var = "gender",
  group_mappings = list("M" = 0, "F" = 1),
  cols_to_remove = c("subjectid")
)
```

train_gnn	<i>Train and Evaluate the Gated Neural Network (robust splits + safe ROC)</i>
-----------	---

Description

Train and Evaluate the Gated Neural Network (robust splits + safe ROC)

Usage

```
train_gnn(
  prepared_data,
  hyper_grid,
  num_repeats = 20,
  epochs = 300,
  output_dir = tempdir(),
  run_tuning = TRUE,
  best_params = NULL,
  save_outputs = FALSE,
```

```
    seed = NULL,  
    verbose = FALSE  
  )
```

Arguments

<code>prepared_data</code>	List from <code>prepare_data()</code> : <code>X</code> , <code>y</code> , <code>group</code> , <code>feature_names</code> , <code>subject_ids</code>
<code>hyper_grid</code>	<code>data.frame</code> with columns: <code>lr</code> , <code>hidden_dim</code> , <code>dropout_rate</code> , <code>lambda</code> , <code>temperature</code>
<code>num_repeats</code>	Integer, repeated train/test splits per combo & final run
<code>epochs</code>	Integer, epochs per run
<code>output_dir</code>	Directory to write csv/rds (defaults to <code>tempdir()</code>)
<code>run_tuning</code>	Logical, run hyperparameter search
<code>best_params</code>	<code>data.frame</code> /list with <code>lr</code> , <code>hidden_dim</code> , <code>dropout_rate</code> , <code>lambda</code> , <code>temperature</code> if <code>run_tuning=FALSE</code>
<code>save_outputs</code>	Logical, whether to save outputs to disk (default <code>FALSE</code>)
<code>seed</code>	Optional seed for reproducible data splits. Defaults to <code>NULL</code> to respect the current RNG state.
<code>verbose</code>	Logical, whether to print progress messages (default <code>FALSE</code>)

Value

`list`(`final_results`, `gate_weights`, `expert_weights`, `performance_summary`, `aif360_data`, `tuning_results`)

Index

`analyse_experts`, [2](#)
`analyse_gnn_results`, [3](#)
`analyse_gnn_results_plus`, [4](#)

`plot_sankey`, [4](#)
`prepare_data`, [5](#)

`train_gnn`, [6](#)