netDx-manual Documentation

Release latest

Jan 29, 2020

Contents

The first version of the netDx user manual is in active development and not all sections are currently complete. Please contact Shraddha Pai at shraddha.pai@utoronto.ca if these pages do not answer your questions.

Citation: Pai S., Hui S., Isserlin R., Shah M.A., Kaka H., and GD Bader. (2019) netDx: Interpretable patient classification using integrated patient similarity networks]. *Molecular Systems Biology* **15**, e8497. [Publication]

The netDx software is available at: https://github.com/BaderLab/netDx

Contents:

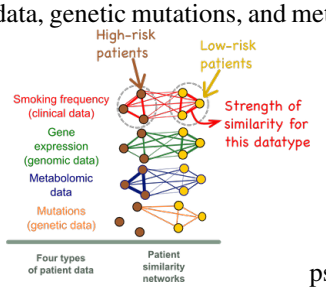
CHAPTER 1

Introduction

netDx is a **patient classifier** algorithm that can integrate several types of patient data into a single model. It specializes in the use of genomic data, which is distinct in the number and correlation structure of measures (e.g. 20,000 genes)It does this by converting each type of data into a view of patient similarity; i.e. by converting the data into a graph in which more similar patients are tightly linked, while less similar patients are not so tightly linked.

1.1 Motivation

In this example, we try to predict which patients are at high-risk for lung cancer. We have four types of data: relevant clinical variables, including smoking frequency, gene expression data, genetic mutations, and metabolomic data. netDx converts the data into 4 views of patient similarity (edge strength

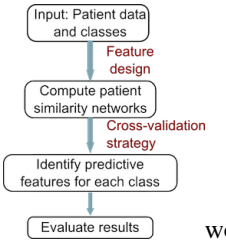


psn_intro.png

In the graphs above, the nodes are patients and the edges are weighted by similarity for that particular datatype. It is evident that the high-risk patients form a strongly interconnected cluster based on smoking frequency (red network) but that the clustering is less evident for gene expression data (green network).

1.2 How netDx works

The conceptual workflow for netDx is shown below. netDx starts with patient data as above. It allows users to define similarity for each of the input datatypes and creates the resulting patient similarity networks. It then uses machine learning to identify which of the input features were predictive for each class. Finally, it uses the predictive features to classify new patients of unknown type.



workflow.png

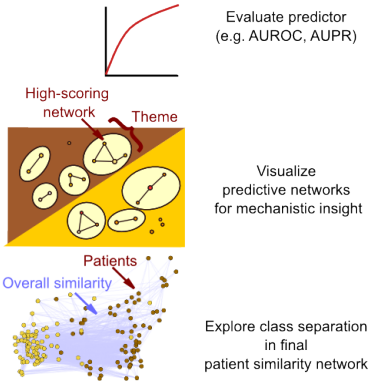
An important aspect of the predictor is the score associated with each input feature. This score indicates the frequency with which cross-validation identified a particular network as predictive for a patient label, and is a measure of predictive power. A threshold can be applied to this score, making passing networks "feature-selected".

1.3 Output

netDx broadly has two purposes. First, it serves as a classifier that can integrate heterogeneous datatypes. Second, it serves as a tool for clinical discovery and research, as identified features may provide mechanistic insight into the condition under study or identify new biomarkers.

netDx therefore provides several types of output that allow the user to examine the nature of the predictor:

- **Predicted labels** for test patients. If nested cross-validation is used, labels for all iterations are provided, along with individual-level classification accuracy.
- **Summary network scores:** Network-level scores for all cross-validation folds. Applying a cutoff for these results in "feature-selected" networks.
- Detailed output: All **intermediate results**, showing network rankings across cross-validation
- An **overall patient similarity network** created by integrating feature-selected networks
- Where applicable, a network visualization of selected features (also called an EnrichmentMap) is generated. This view shows the major themes present in feature-selected variables.



outputs.png

CHAPTER 2

Installation

Detailed install instructions are provided on the netDx github repo page.

CHAPTER 3

Create Patient Similarity Networks (or Feature Design)

3.1 Overview

Defining meaningful patient similarity networks (or features) is the key to building a high-performing classifier. It is also crucial to using netDx for clinical discovery. When designing patient similarity networks, two considerations are the level at which individual variables should be grouped, and the measure used to define patient similarity.

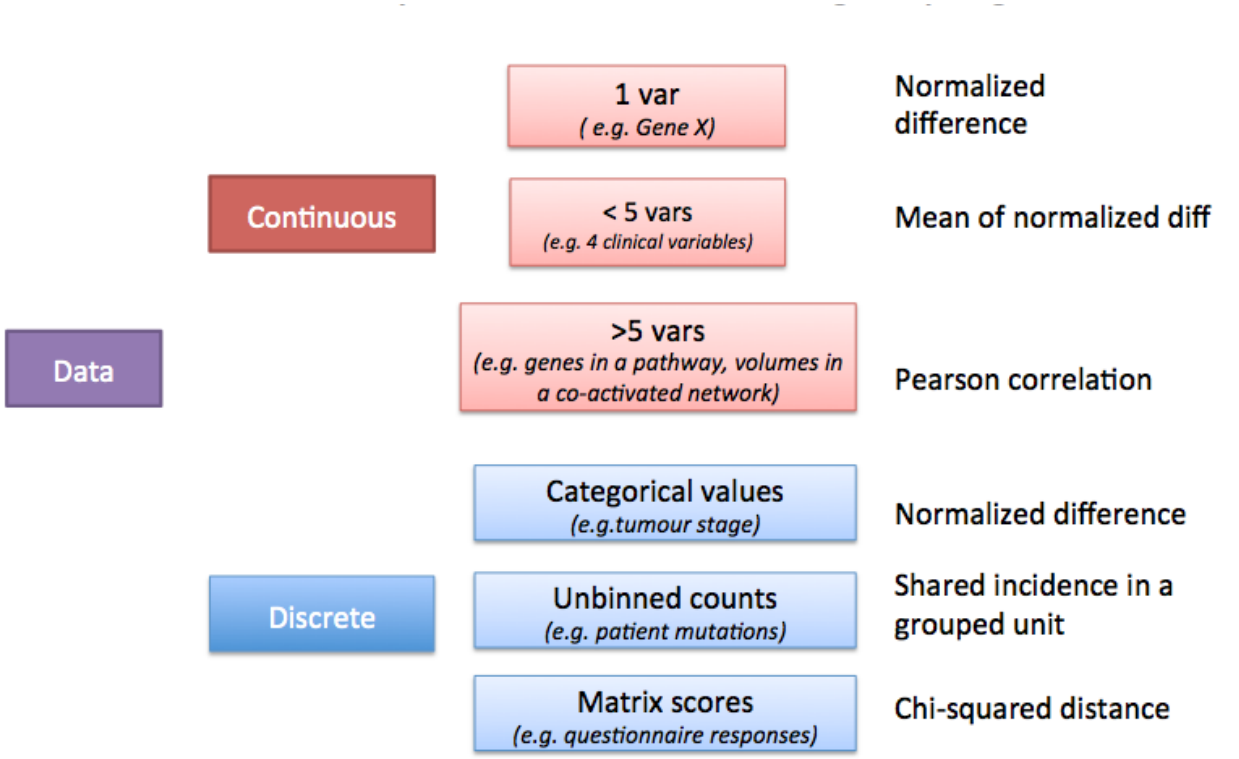
3.2 Grouping Variables: What makes an input network?

In a given datatype, not all measured variables are equally predictive. Moreover some groupings of variables may be more informative about mechanism, perhaps because they use prior knowledge. Such variables are more interpretable, as they reflect some process of clinical or biological relevance. For each datatype, the user needs to decide which level of variable-grouping to apply.

This table provides some examples for different types of data. Consider the lung cancer example from the *Introduction*, and assume we measure: 34 clinical variables; 20,000 genes; 16 metabolites; and genetic mutations from whole-genome sequencing.

3.3 Choosing a similarity metric

Patient similarity can be measured in different ways for different types of input data. Here is a set of recommended metrics to start with, depending on what the type of data is and how many variables were grouped to create a given net.



sim_metrics.png

3.4 Summary table

3.4.1 Defining custom similarity functions

netDx is agnostic to the choice of similarity metric. Set the `simMetric` argument of `makePSN_NamedMatrix` to `custom` to provide a user-defined similarity metric. Be aware that the choice of similarity metric could increase false positives or false negatives during feature selection. Build controls to guard against these.

3.5 Expression data

This situation applies to a table of >5 values with continuous-valued measures. An example is a table of gene expression data, with ~20,000 measures per patient. Another is proteomic data with ~20 measures per patient.

Suggested metric: *Pearson correlation*. This is the default similarity metric for `makePSN_NamedMatrix()` so no special specification is required.

Note: Be sure to set `writeProfiles=TRUE`.

3.6 Fewer than 5 datapoints

Pearson correlation is not a stable measure of similarity when there are fewer than 5 variables per patient. An alternate similarity measure in such a situation is the **average normalized difference for each variable**.

$$S(a,b,G) = \frac{\sum_{i=1}^{k} \frac{abs(a_i - b_i)}{max(g_i) - min(g_i)}}{k}$$

avg_normDiff

```
#' Similarity by average of normalized difference
#'
#' @details Similarity measure for network with 2-5 variables.
#' Defined as the average of normalized difference for each of the
#' member variables
#' @param x (matrix) rows are patients, columns are values for component
#' variables
normDiff_avg <- function(x) {

        # normalized difference for a single variable
        normDiff <- function(x) {
            nm <- colnames(x)
            x <- as.numeric(x)
            n <- length(x)
            rngX  <- max(x,na.rm=T)-min(x,na.rm=T)

            out <- matrix(NA,nrow=n,ncol=n);
            # weight between i and j is
            # wt(i,j) = 1 - (abs(x[i]-x[j])/(max(x)-min(x)))
            for (j in 1:n) out[,j] <- 1-(abs((x-x[j])/rngX))
            rownames(out) <- nm; colnames(out)<- nm
            out
        }

        sim <- matrix(0,nrow=ncol(x),ncol=ncol(x))
        for (k in 1:nrow(x)) {
                tmp <- normDiff(x[k,,drop=FALSE])
                sim <- sim + tmp
                rownames(sim) <- rownames(tmp)
                colnames(sim) <- colnames(tmp)
        }
        sim <- sim/nrow(x)
        sim
}
```

Use in `netDx`: Given

- `dat`matrix with 5 variables (5xN matrix, where N is number of patients)
- `dat_names` vector with variable names
- `myGroup` list with the group name as key and members as variables,

then the call to create networks would be:

```
makePSN_NamedMatrix(dat,dat_names,myGroup,
                        outDir,simMetric="custom",customFunc=normDiff_avg,
                        sparsify=TRUE)
```

Note:

- `simMetric="custom"`

- `customFunc` points to the custom function definition
- `writeProfiles=FALSE`
- `sparsify=TRUE` *keep only the strongest edges for efficient memory use*

3.7 Range-based data (genetic mutations)

Creating patient data from genomic events such as genetic mutations or DNA copy number polymorphisms, requires a different design for creating PSN.

This section coming soon - 170927

3.8 Setting up to get an enrichment map

CHAPTER 4

Design the Predictor

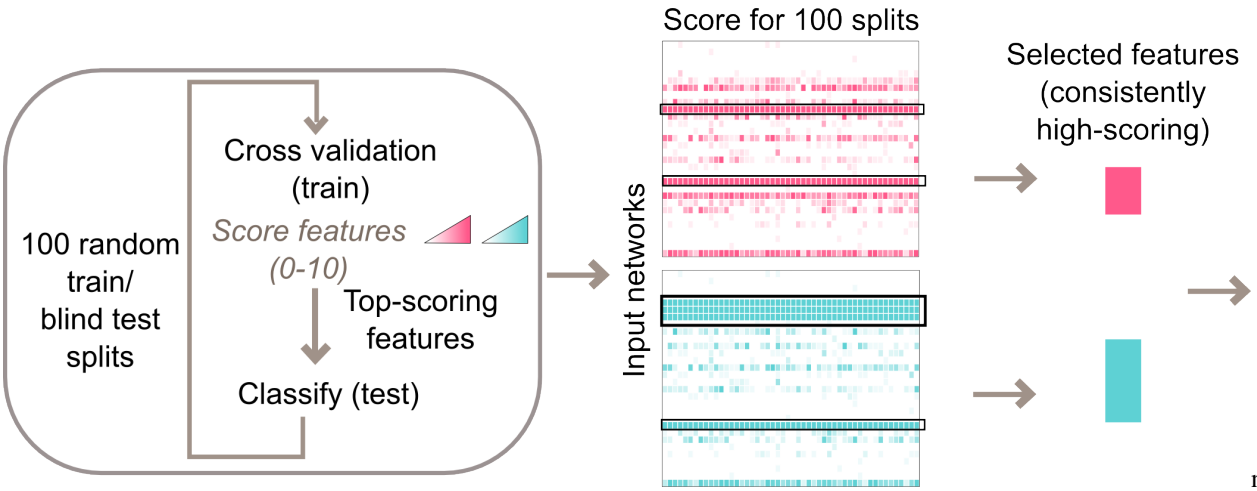
4.1 Nested cross-validation design

We recommend a nested cross-validation design for predictor building. **Inner loop:** Patient samples are split (e.g. 80:20) into a training and a blind test set. Using only the training samples, 10-fold cross validation is performed for each class, generating for each network a score between 0 and 10. Networks that scored 9 or 10 out of 10 are used to classify blind test samples. Note that this loop also generates a performance score for this particular blind test sample.

Outer loop: The inner loop is repeated (e.g. 100 times), which each loop using a new random split of train and blind test (100 trials).

The average of blind test classification across the 100 splits is used to measure overall predictor performance.

Feature-selected networks are those that score consistently well across all the outer loops. For example, a strict criterion would require networks to score score 10 out of 10 in all loops.



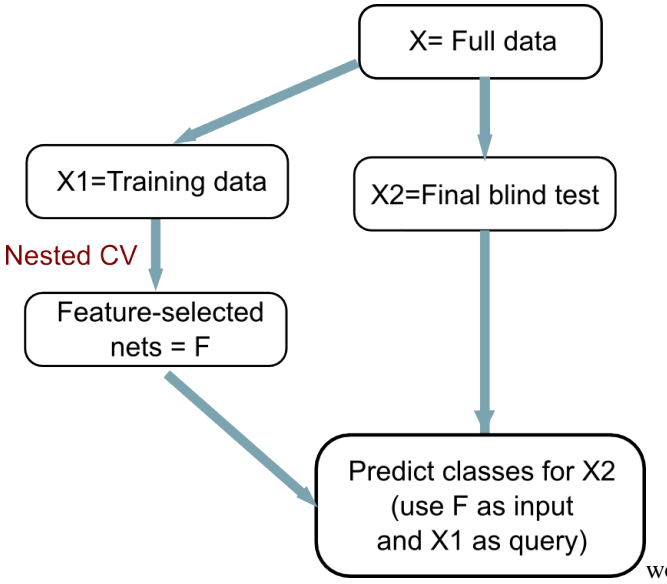
nestedCV.png

Example of nested cross-validation design in binary classification. Here an inner loop of 10-fold cross validation is run 100 times over different splits of data into train and blind test. At the end of the process, each class has a **network**

score table. This matrix (call it `S`) has K rows (where K is the number of networks) and 100 columns. `S[i,j]` has the cross-validation score for network `i` in loop `j`. The user then applies a cutoff to call **feature-selected networks**.

4.2 Overall workflow

Ideally a dataset would have sufficient samples to be set aside at the outset, before nested CV (figure below). Comparing predictor performance from the nested CV to that with the final blind test serves as a validation test.



workflow_predictor.png

CHAPTER 5

Indices and tables

- genindex
- modindex
- search