Genetics and Bioinformatics GBIO0002

Archana Bhardwaj

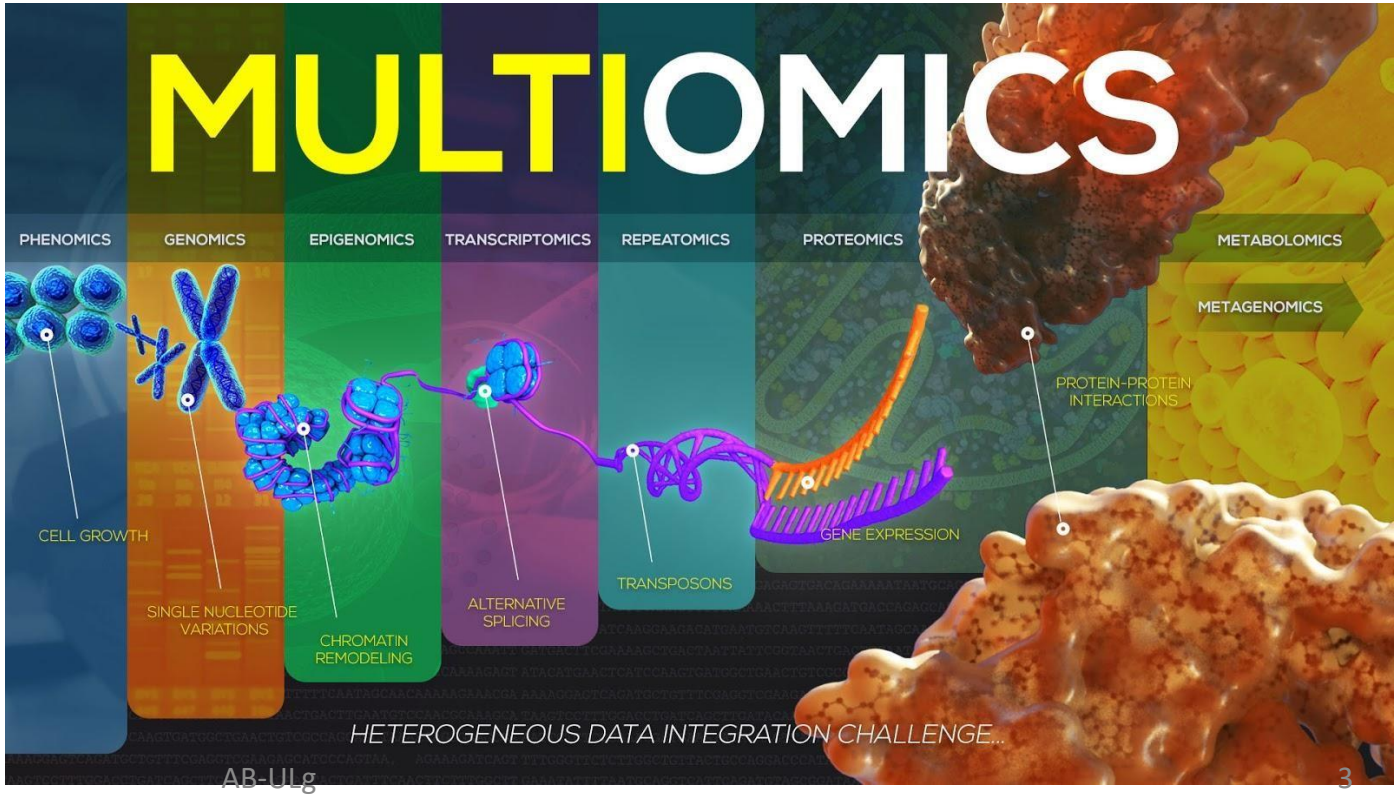
Omics integration

Goal of session

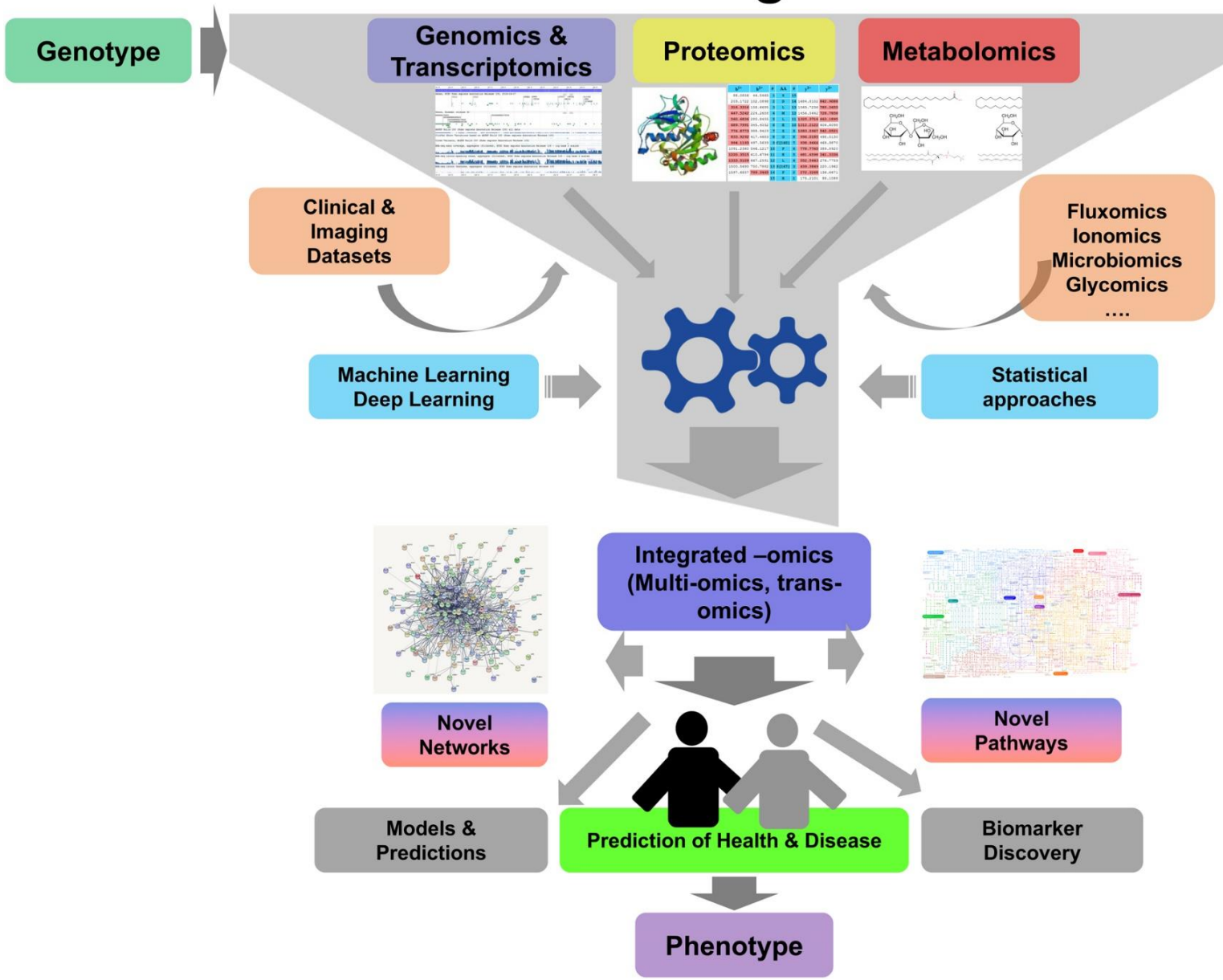
1. Develop the Patient-Patient interaction network

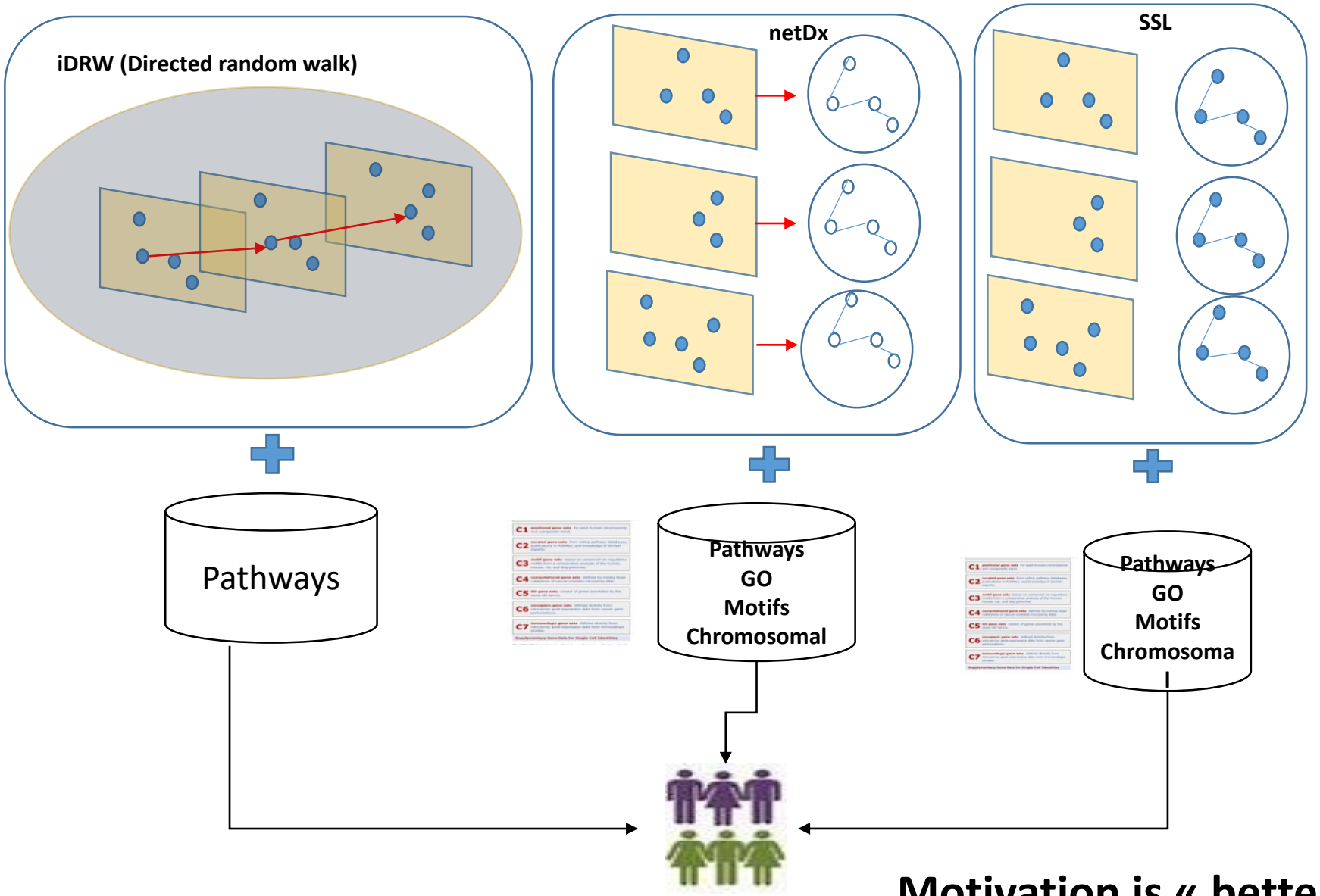
2. Develop Gene-Gene interaction network

Multiomics, multi-omics or integrative omics is a biological analysis approach in which the data sets are multiple "omes", such as the genome, proteome, transcriptome, epigenome, and microbiome; in other words, the use of multiple omics technologies to study life in a concerted way.



Workflows in Integrated Omics





Multiple approches are available

1. **IDRW**
2. **NetDX**
3. **SSL**

Making use of biological knowledge

Motivation is « better Clinical outcome »

1.Develop the Patient-Patient interaction network

Biological Question need to be addressed

- **What are possible biological entities making difference among two group of cancer patients ?**
- **Can we use multiple data types ?**

NetDx

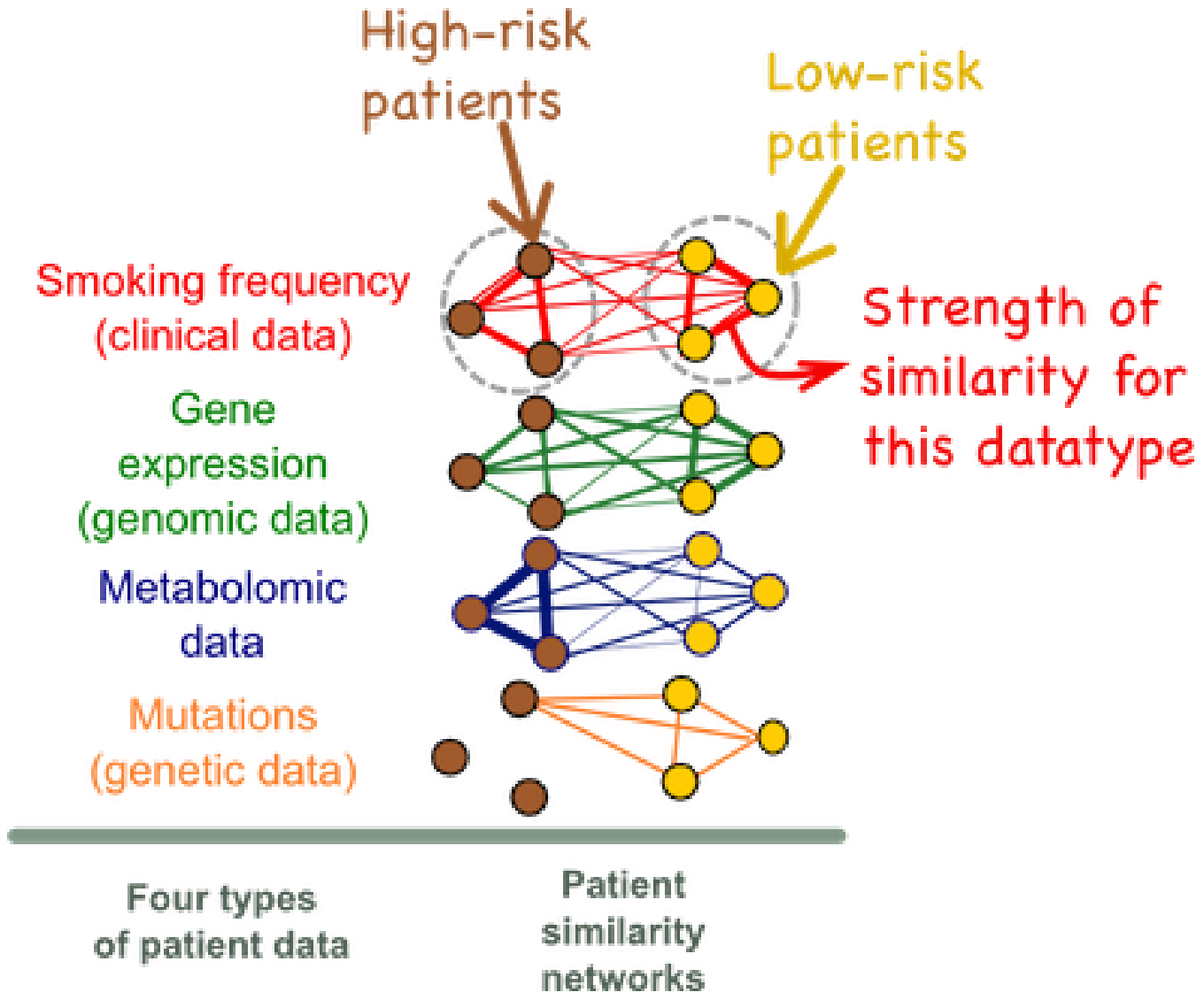
- One can predict which patients are at high-risk for specific cancer or not

- One can deal with multiple data types : relevant clinical variables, including smoking frequency, gene expression data, genetic mutations, and metabolomic data.

- netDx converts the data into 4 views of patient similarity

netDx

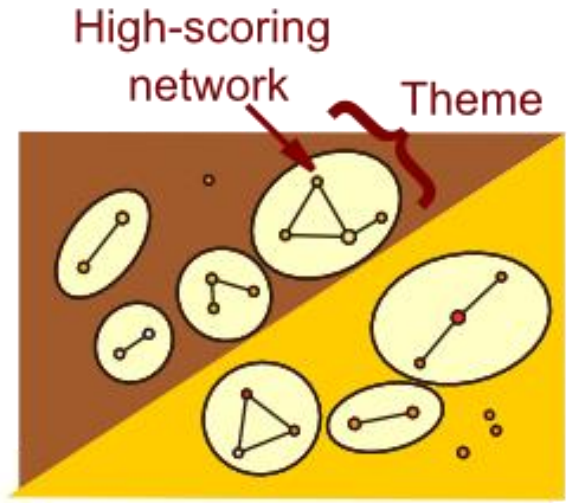
- The high-risk patients form a strongly interconnected cluster based on smoking frequency (red network) but that the clustering is less evident for gene expression data (green network).
- The nodes are patients and the edges are weighted by similarity for that particular datatype.



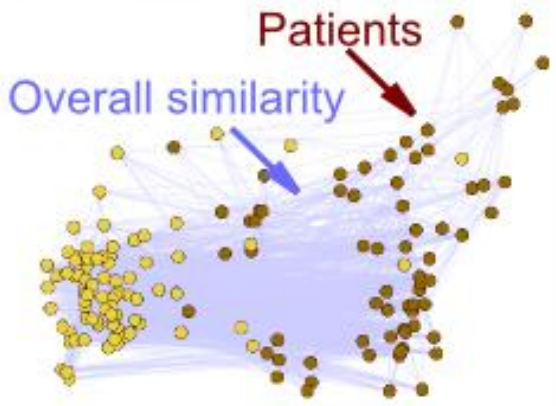
Motivation to use « netDx »

- **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.**

Evaluate predictor
(e.g. AUROC, AUPR)



Visualize
predictive networks
for mechanistic insight



Explore class separation
in final
patient similarity network

HOW netDx works ?

- **netDx starts with patient data**

- **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".**

- **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.**

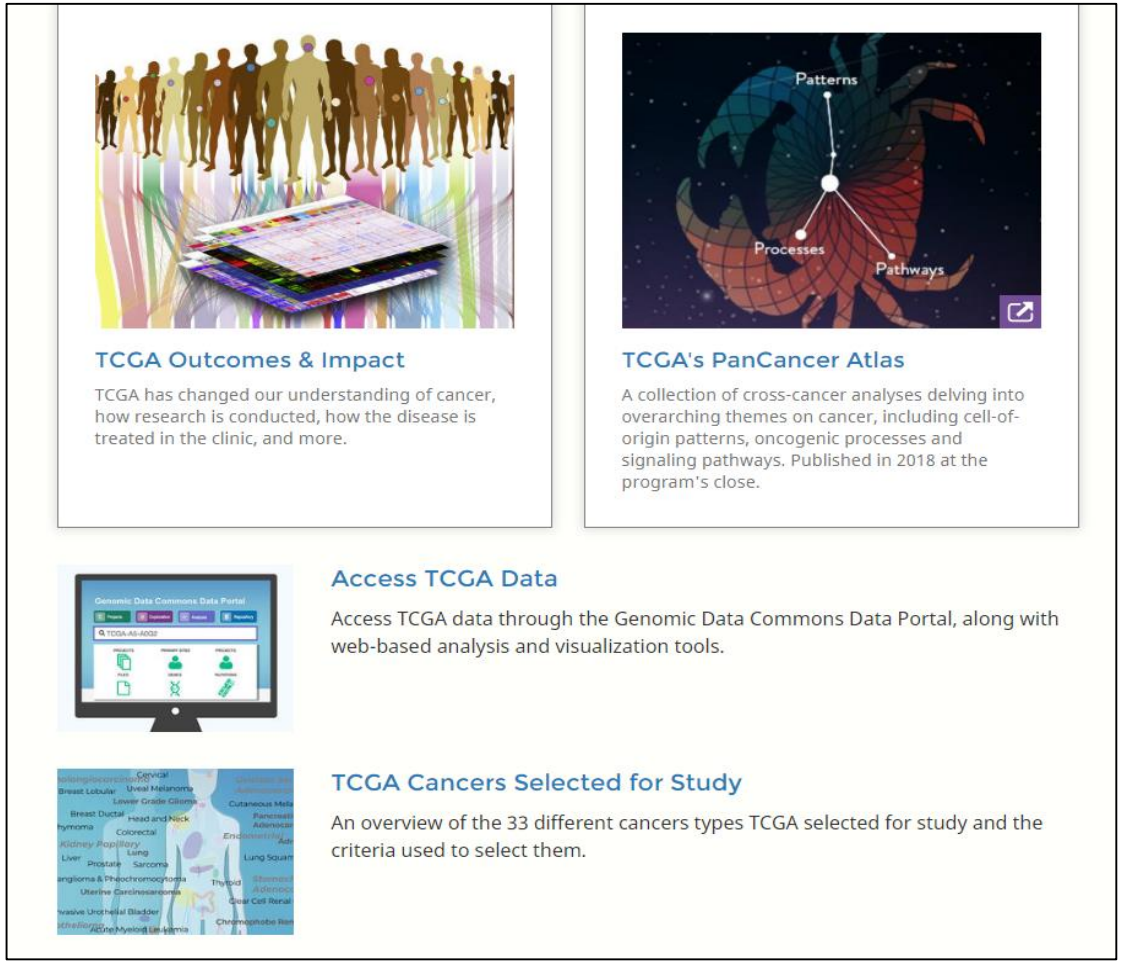
Installation

- To download file, got to link https://github.com/BaderLab/netDx
- click on " install netDx v1.0.23“
- Uncompress folder and follow instructions given below

```
$ cd netDx/
$ R
> install.packages(c('devtools','curl'))
>install.packages(c("bigmemory","foreach","combinat","doParallel","ROC
R","pracma","RColorBrewer","reshape2","ggplot2","tinytex","rmarkdown
","caroline","glmnet","igraph","knitr"))
> BiocManager::install(c("GenomicRanges","RCy3"))
> install.packages("netDx",type="source",repos=NULL)
> install.packages("netDx.examples",type="source",repos=NULL)
```

TCGA DATABASE

The Cancer Genome Atlas (TCGA), a landmark cancer genomics program, molecularly characterized over 20,000 primary cancer and matched normal samples spanning 33 cancer types. This joint effort between the National Cancer Institute and the National Human Genome Research Institute began in 2006, bringing together researchers from diverse disciplines and multiple institutions.



netDx : Input

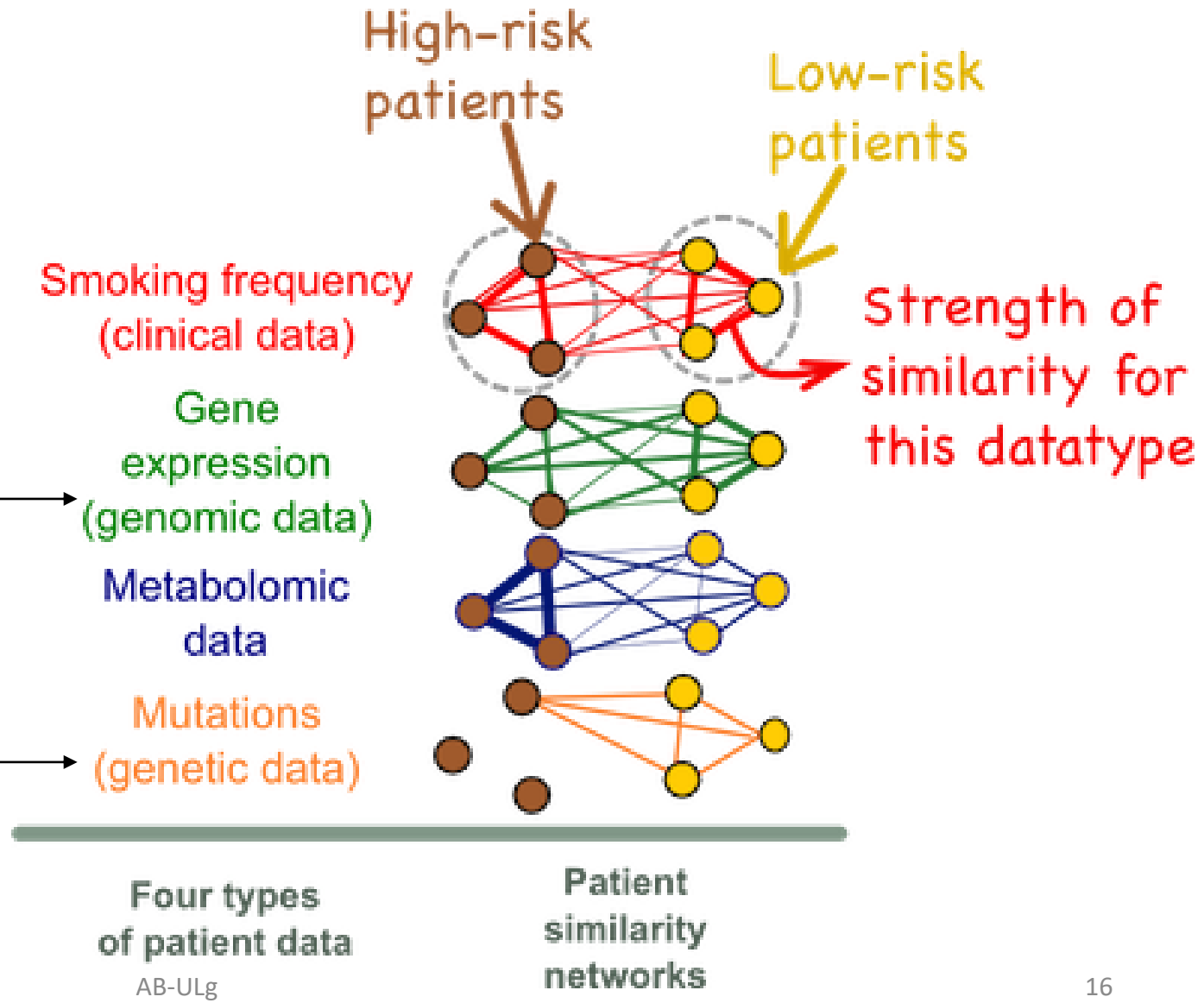
: Smokers	% Current Smok
65	13.13
20	23.28
41	17.52
25	35.85
23	34.11
46	28.38
46	23.33

Genes

	sample1	sample2	sample3
1	0.46	0.30	0.80
2	-0.10	0.49	0.24
3	0.15	0.74	0.04
4	-0.45	-1.03	-0.79
5	-0.06	1.06	1.35

Genes

sample1	sample2	sample3
1	0	0
0	0	0
0	1	1
0	0	0
0	0	1
0	0	1
1	0	0
0	0	0
0	0	0
0	0	0
0	0	0



netDx : Output

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

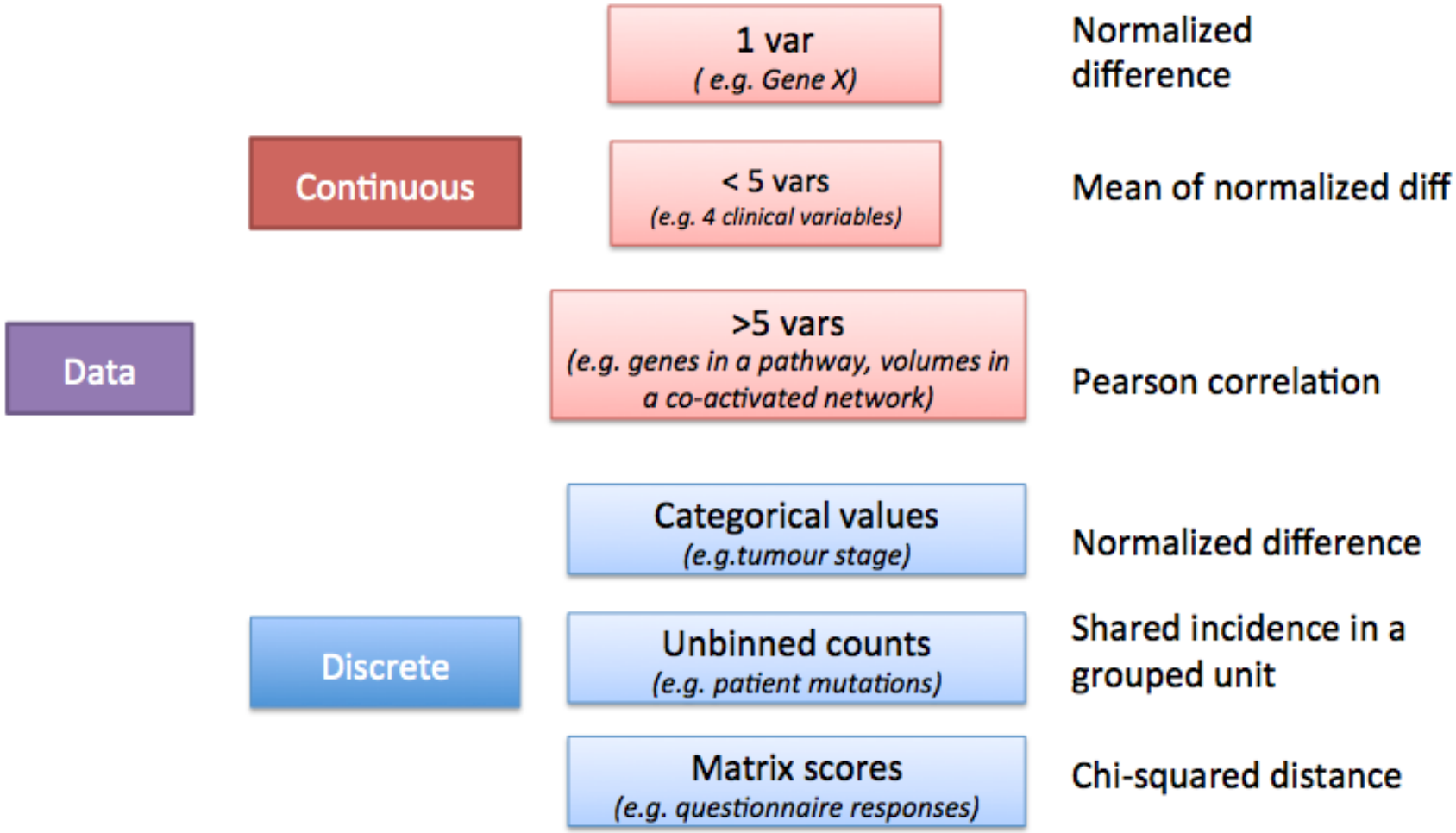
- Predicted labels for test patients. If nested cross-validation is used, labels for all iterations are provided, along with individual-level classification accuracy.

- 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**

Custom Functions based on data types



netDx takes custom similarity functions for provided input

Exercise Outcome

- **Perform feature selection on the training set**
- **Assess performance on the test set**
- **Generate patient similarity networks from more than one type of data**

CANCER TYPE IN TCGA

Cancer Types

Select a type of cancer to learn about treatment, causes and prevention, screening, and the latest research.

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

A

Acute Lymphoblastic Leukemia (ALL)
Acute Myeloid Leukemia (AML)
Adolescents, Cancer in
Adrenocortical Carcinoma
Childhood Adrenocortical Carcinoma - see Unusual Cancers of Childhood
AIDS-Related Cancers
Kaposi Sarcoma (Soft Tissue Sarcoma)
AIDS-Related Lymphoma (Lymphoma)
Primary CNS Lymphoma (Lymphoma)
Anal Cancer
Appendix Cancer - see Gastrointestinal Carcinoid Tumors
Astrocytomas, Childhood (Brain Cancer)
Atypical Teratoid/Rhabdoid Tumor, Childhood, Central Nervous System (Brain Cancer)

B

Common Cancer Types

Bladder Cancer
Breast Cancer
Colon and Rectal Cancer
Endometrial Cancer
Kidney Cancer
Leukemia
Liver Cancer
Lung Cancer
Melanoma
Non-Hodgkin Lymphoma
Pancreatic Cancer
Prostate Cancer
Thyroid Cancer

- **We will download (TCGA-BRCA) data for today session**

Data Preparation

```
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")

BiocManager::install("MultiAssayExperiment")

BiocManager::install("curatedTCGAData")
```

MultiAssayExperiment harmonizes data management of multiple assays performed on an overlapping set of specimens. It provides a familiar Bioconductor user experience by extending concepts from SummarizedExperiment, supporting an open-ended mix of standard data classes for individual assays, and allowing subsetting by genomic ranges or rownames.

Let Us Download Cancer Data

```
library(curatedTCGAData)
library(MultiAssayExperiment)
curatedTCGAData(diseaseCode="BRCA", assays="*",dru.run=TRUE)
```

```
> curatedTCGAData(diseaseCode="BRCA", assays="*",dru.run=TRUE)
                                          Title DispatchClass
31                        BRCA_CNASeq-20160128           Rda
32                        BRCA_CNASNP-20160128           Rda
33                        BRCA_CNVSNP-20160128           Rda
35             BRCA_GISTIC_AllByGene-20160128           Rda
36                 BRCA_GISTIC_Peaks-20160128           Rda
37     BRCA_GISTIC_ThresholdedByGene-20160128           Rda
39  BRCA_Methylation_methyl27-20160128_assays        H5File
40      BRCA_Methylation_methyl27-20160128_se           Rds
41 BRCA_Methylation_methyl450-20160128_assays        H5File
42     BRCA_Methylation_methyl450-20160128_se           Rds
43                 BRCA_miRNASeqGene-20160128           Rda
44                    BRCA_mRNAArray-20160128           Rda
45                     BRCA_Mutation-20160128           Rda
46             BRCA_RNASeq2GeneNorm-20160128           Rda
47                  BRCA_RNASeqGene-20160128           Rda
48                    BRCA_RPPAArray-20160128           Rda
>
```

For each disease type, one need to give specific Disease code

Let us create Multi Assay Experiment – I

```
brca <- curatedTCGAData("BRCA",c("mRNAArray","Mutation"),FALSE)
```

```
> brca <- curatedTCGAData("BRCA",c("mRNAArray","Mutation"),FALSE)
  |======================================================================================================| 100%

snapshotDate(): 2019-10-22
see ?curatedTCGAData and browseVignettes('curatedTCGAData') for documentation
loading from cache
see ?curatedTCGAData and browseVignettes('curatedTCGAData') for documentation
loading from cache
Loading required package: RaggedExperiment
see ?curatedTCGAData and browseVignettes('curatedTCGAData') for documentation
loading from cache
see ?curatedTCGAData and browseVignettes('curatedTCGAData') for documentation
loading from cache
see ?curatedTCGAData and browseVignettes('curatedTCGAData') for documentation
loading from cache
harmonizing input:
  removing 12790 sampleMap rows not in names(experiments)
  removing 104 colData rownames not in sampleMap 'primary'
>
```

We will work with two omics profiles : mRNA and Mutation

Let us create Multi Assay Experiment – II

```
> brca
```

```
> brca
A MultiAssayExperiment object of 2 listed
 experiments with user-defined names and respective classes.
 Containing an ExperimentList class object of length 2:
 [1] BRCA_mRNAArray-20160128: SummarizedExperiment with 17814 rows and 590 columns
 [2] BRCA_Mutation-20160128: RaggedExperiment with 90490 rows and 993 columns
Features:
 experiments() - obtain the ExperimentList instance
 colData() - the primary/phenotype DataFrame
 sampleMap() - the sample availability DFrame
 `$`, `[`, `[[` - extract colData columns, subset, or experiment
 *Format() - convert into a long or wide DataFrame
 assays() - convert ExperimentList to a SimpleList of matrices
```

Here , 2 list indicates Mutation and mRNA data

Let us create Multi Assay Experiment – III

```
pID <- colData(brca)$patientID
```

```
> pID
  [1] "TCGA-A1-A0SB" "TCGA-A1-A0SD" "TCGA-A1-A0SE" "TCGA-A1-A0SF" "TCGA-A1-A0SG" "TCGA-A1-A0SH" "TCGA-A1-A0SI"
  [8] "TCGA-A1-A0SJ" "TCGA-A1-A0SK" "TCGA-A1-A0SM" "TCGA-A1-A0SN" "TCGA-A1-A0SO" "TCGA-A1-A0SP" "TCGA-A1-A0SQ"
 [15] "TCGA-A2-A04N" "TCGA-A2-A04P" "TCGA-A2-A04Q" "TCGA-A2-A04R" "TCGA-A2-A04T" "TCGA-A2-A04U" "TCGA-A2-A04V"
 [22] "TCGA-A2-A04W" "TCGA-A2-A04X" "TCGA-A2-A04Y" "TCGA-A2-A0CK" "TCGA-A2-A0CL" "TCGA-A2-A0CM" "TCGA-A2-A0CO"
 [29] "TCGA-A2-A0CP" "TCGA-A2-A0CQ" "TCGA-A2-A0CR" "TCGA-A2-A0CS" "TCGA-A2-A0CT" "TCGA-A2-A0CU" "TCGA-A2-A0CV"
 [36] "TCGA-A2-A0CW" "TCGA-A2-A0CX" "TCGA-A2-A0CY" "TCGA-A2-A0CZ" "TCGA-A2-A0D0" "TCGA-A2-A0D1" "TCGA-A2-A0D2"
 [43] "TCGA-A2-A0D3" "TCGA-A2-A0D4" "TCGA-A2-A0EM" "TCGA-A2-A0EN" "TCGA-A2-A0EO" "TCGA-A2-A0EP" "TCGA-A2-A0EQ"
 [50] "TCGA-A2-A0ER" "TCGA-A2-A0ES" "TCGA-A2-A0ET" "TCGA-A2-A0EU" "TCGA-A2-A0EV" "TCGA-A2-A0EW" "TCGA-A2-A0EX"
 [57] "TCGA-A2-A0EY" "TCGA-A2-A0ST" "TCGA-A2-A0SU" "TCGA-A2-A0SV" "TCGA-A2-A0SW" "TCGA-A2-A0SX" "TCGA-A2-A0SY"
 [64] "TCGA-A2-A0T0" "TCGA-A2-A0T1" "TCGA-A2-A0T2" "TCGA-A2-A0T3" "TCGA-A2-A0T4" "TCGA-A2-A0T5" "TCGA-A2-A0T6"
 [71] "TCGA-A2-A0T7" "TCGA-A2-A0YC" "TCGA-A2-A0YD" "TCGA-A2-A0YE" "TCGA-A2-A0YF" "TCGA-A2-A0YG" "TCGA-A2-A0YH"
 [78] "TCGA-A2-A0YI" "TCGA-A2-A0YJ" "TCGA-A2-A0YK" "TCGA-A2-A0YL" "TCGA-A2-A0YM" "TCGA-A2-A0YT" "TCGA-A2-A1FV"
 [85] "TCGA-A2-A1FW" "TCGA-A2-A1FX" "TCGA-A2-A1FZ" "TCGA-A2-A1G0" "TCGA-A2-A1G1" "TCGA-A2-A1G4" "TCGA-A2-A1G6"
 [92] "TCGA-A2-A259" "TCGA-A2-A25A" "TCGA-A2-A25B" "TCGA-A2-A25C" "TCGA-A2-A25D" "TCGA-A2-A25E" "TCGA-A2-A25F"
 [99] "TCGA-A2-A3KC" "TCGA-A2-A4RW" "TCGA-A2-A4RY" "TCGA-A2-A4S2" "TCGA-A7-A0CD" "TCGA-A7-A0CE" "TCGA-A7-A0CG"
[106] "TCGA-A7-A0CH" "TCGA-A7-A0CJ" "TCGA-A7-A0D9" "TCGA-A7-A0DA" "TCGA-A7-A0DB" "TCGA-A7-A0DC" "TCGA-A7-A13D"
[113] "TCGA-A7-A13E" "TCGA-A7-A13F" "TCGA-A7-A13G" "TCGA-A7-A13H" "TCGA-A7-A26E" "TCGA-A7-A26F" "TCGA-A7-A26G"
[120] "TCGA-A7-A26H" "TCGA-A7-A26I" "TCGA-A7-A26J" "TCGA-A7-A3IZ" "TCGA-A7-A3J1" "TCGA-A7-A426" "TCGA-A7-A4SA"
```

```
> length(pID)
[1] 994
```

We will work with 994 patients samples

Samples Detail - I

```
pam50 <- colData(brca)$PAM50.mRNA
```

```
> pam50
  [1] NA              "Luminal A"     "Luminal A"     NA              NA              "Luminal A"
  [7] NA              "Luminal A"     "Basal-like"    "Luminal B"     NA              "Basal-like"
 [13] NA              NA              "Luminal A"     "Basal-like"    "Basal-like"    "Luminal B"
 [19] "Basal-like"    "Basal-like"    "Luminal A"     "HER2-enriched" "HER2-enriched" "Luminal A"
 [25] NA              "HER2-enriched" "Basal-like"    NA              "Luminal A"     "Luminal A"
 [31] NA              "Luminal A"     "Luminal B"     "Luminal A"     "Luminal A"     "Luminal B"
 [37] "HER2-enriched" "HER2-enriched" "Luminal A"     "Basal-like"    "HER2-enriched" "Basal-like"
 [43] "Luminal A"     "Luminal B"     "Luminal A"     "Luminal A"     "Luminal A"     NA
 [49] "HER2-enriched" "Luminal B"     "Luminal A"     "Luminal A"     "Luminal A"     "Luminal A"
 [55] "Luminal A"     "Luminal A"     "Luminal B"     "Basal-like"    "Luminal A"     "Luminal B"
 [61] "Luminal B"     "Basal-like"    "Luminal A"     "Basal-like"    "HER2-enriched" "Basal-like"
 [67] "Luminal B"     "Luminal B"     "Luminal A"     "Luminal A"     "Luminal A"     "Luminal A"
 [73] "Luminal A"     "Basal-like"    "Luminal A"     "Luminal B"     "Luminal B"     "Luminal A"
 [79] "Basal-like"    "Normal-like"   "Luminal A"     "Basal-like"    NA              NA
 [85] NA              NA              NA              NA              NA              NA
 [91] NA              NA              NA              NA              NA              NA
```

Each sample belongs to different group such as Luminal type, Basal Like and others.

Samples Detail – II

```
staget <- colData(brca)$pathology_T_stage
```

```
st2 <- rep(NA,length(staget))
st2[which(staget %in% c("t1","t1a","t1b","t1c"))] <- 1
st2[which(staget %in% c("t2","t2a","t2b"))] <- 2
st2[which(staget %in% c("t3","t3a"))] <- 3
st2[which(staget %in% c("t4","t4b","t4d"))] <- 4
colData(brca)$STAGE <- st2
```

```
pam50[which(!pam50 %in% "Luminal A")] <- "notLumA"
pam50[which(pam50 %in% "Luminal A")] <- "LumA"
colData(brca)$ID <- pID
colData(brca)$STAGE <- st2
colData(brca)$STATUS <- pam50
```

Add samples tags as notLumA and LumA

Extract colData of samples groups as notLumA and LumA

Work with tumour samples

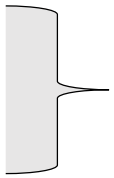
```
idx <- union(which(pam50 == "Normal-like"), which(is.na(st2)))
cat(sprintf("excluding %i samples\n", length(idx)))
```

excluding 2 samples

```
tokeep <- setdiff(pID, pID[idx])
brca <- brca[,tokeep,]
```

Work with Tumour samples

```
dim(colData(brca))
[1]  992 2687
```



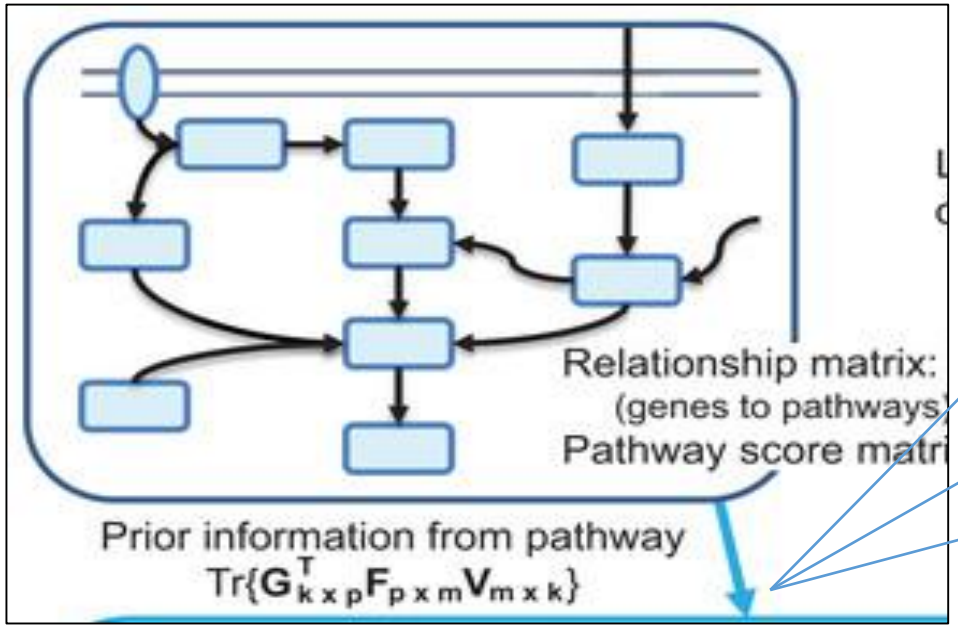
No of samples reduced from 994 to 992

NetDx : Background Prior Knowledge

- **NetDx uses prior knowledge of pathways.**
- **One can change prior knowledge of pathways with cancer gene sets , immunogeneic gene signatures and many others.**

Prior Biological knowledge

- There are many knowledge in Bio : pathway databases [6–8], Gene Ontology [9] and others.
- So, Integrating known information from databases and biological literature as prior knowledge thus appears to be beneficial.




```
$GUANOSINE_NUCLEOTIDES__I_DE_NOVO__I__BIOSYNTHESIS
 [1] "NME7"  "NME6"  "RRM2B" "GMPS"  "NME2"  "NME3"  "NME4"  "NME5"
 [9] "RRM2"  "NME1"  "GUK1"  "RRM1"  "IMPDH2" "IMPDH1"

$RETINOL_BIOSYNTHESIS
 [1] "RDH10" "DHRS4" "LRAT"  "LIPC"  "CES5A" "DHRS9" "RDH11" "DHRS3" "CES1"
[10] "RBP1"  "CES4A" "RBP2"  "PNLIP" "RBP5"  "RBP4"  "CES2"

$`MUCIN_CORE_1_AND_CORE_2__I_O__I_-GLYCOSYLATION`
 [1] "GALNT1"  "GCNT4"   "GALNT7"  "GCNT3"   "GCNT7"   "GALNT6"  "GALNT4"
 [8] "GALNT5"  "ST3GAL2" "ST3GAL1" "ST3GAL4" "GALNT10" "GALNT15" "GALNTL6"
[15] "B3GNT3"  "GALNT16" "GALNT18" "GALNT11" "GALNT12" "GCNT1"   "C1GALT1"
[22] "GALNT13" "GALNT14" "WBSCR17" "GALNT8"  "GALNT9"  "GALNT2"  "GALNT3"
```

Prior Knowledge : Pathways

```
pathList <- readPathways(getExamplePathways())
```

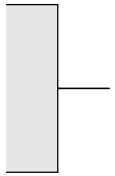
```
> pathList <- readPathways(getExamplePathways())
----------------------------------------

File: 1a78170f7da5_Human_AllPathways_January_24_2016_symbol.gmt


Read 2760 pathways in total, internal list has 2712 entries
        FILTER: sets with num genes in [10, 200]
          => 1006 pathways excluded
          => 1706 left
```

```
brca <- brca[,,1] # keep only clinical and mRNA data
```

Remove Duplicate Arrays

```
smp <- sampleMap(brca)
samps <- smp[which(smp$assay=="BRCA_mRNAArray-20160128"),]
notdup <- samps[which(!duplicated(samps$primary)),"colname"]
brca[[1]] <- brca[[1]][,notdup]
```

```
> dim(colData(brca))
[1]  525 2687
```



Number of samples reduced from 922 to 525

Create List structures

```
groupList <- list()
groupList[["BRCA_mRNAArray-20160128"]] <- pathList[seq_len(3)]
groupList[["clinical"]] <- list(age="patient.age_at_initial_pathologic_diagnosis",
   stage="STAGE")
```

```
> names(groupList)
[1] "BRCA_mRNAArray-20160128" "clinical"
```

We have mRNA and clincial data

```
> groupList[["clinical"]]
$age
[1] "patient.age_at_initial_pathologic_diagnosis"

$stage
[1] "STAGE"
```

Under clinical data, we have age and stage as clinical features

The goal is to create input networks for all possible predictors, before proceeding to feature selection

Function to create Network profiles

```
makeNets <- function(dataList, groupList, netDir,...) {
  netList <- c()
  # make RNA nets: group by pathway
  if (!is.null(groupList[["BRCA_mRNAArray-20160128"]])) {
  netList <- makePSN_NamedMatrix(dataList[["BRCA_mRNAArray-20160128"]],
          rownames(dataList[["BRCA_mRNAArray-20160128"]]),
          groupList[["BRCA_mRNAArray-20160128"]],
          netDir,verbose=FALSE,
          writeProfiles=TRUE,...)
  netList <- unlist(netList)
  cat(sprintf("Made %i RNA pathway nets\n", length(netList)))
  }

  # make clinical nets,one net for each variable
  netList2 <- c()
  if (!is.null(groupList[["clinical"]])) {
  netList2 <- makePSN_NamedMatrix(dataList$clinical,
    rownames(dataList$clinical),
    groupList[["clinical"]],netDir,
    simMetric="custom",customFunc=normDiff, # custom function
    writeProfiles=FALSE,
    sparsify=TRUE,verbose=TRUE,...)
  }
  netList2 <- unlist(netList2)
  cat(sprintf("Made %i clinical nets\n", length(netList2)))
  netList <- c(netList,netList2)
  cat(sprintf("Total of %i nets\n", length(netList)))
  return(netList)
}
```

The function that generates the networks from submatrices of the gene expression data is makePSN NamedMatrix().

- **Develop network profiles based on gene expression data using function makePSN_NamedMatrix**
- **writeProfiles=TRUE (store files in directory)**

Patient similarity matrix creation

- From gene expression data, we create one network per cellular pathway.
- Similarity between two patients is defined as the Pearson correlation of the expression vector; each network is limited to genes for the corresponding pathway.
- In this case, we are generating “profiles”, or simply writing submatrices corresponding to the pathways (note the writeProfiles=TRUE argument).
- As these profiles will create completely connected networks with (N choose 2) edges, weaker edges will first be pruned for computational feasibility.
- We use GeneMANIA to “sparsify” the networks in the GM createDB() subroutine. Note that netList contains the names of networks, rather than the contents; the profiles are written to profDir. Profile file names end with .profile

Key Feature selection functions

- **Runs the cross-validation with successive GeneMANIA queries**
- **Loops over all network rank files (or NRANK files) and computes the network score**

Rank test patients using trained model

- For each of these classes, create a single GeneMANIA database comprising only of the feature selected nets ;
- This is equivalent to our trained model for each class.
- We rank the similarity of a test patient to each class via a GeneMANIA query;
- The query consists of training samples from the corresponding class.

NetDx : Prediction Run

```
out <- buildPredictor(dataList=brca,groupList=groupList,
makeNetFunc=makeNets, ### custom network creation function
outDir=sprintf("%s/pred_output_new",tempdir()), ## absolute path
numCores=16L,featScoreMax=2L, featSelCutoff=1L,numSplits=2L)
```

- Multiple output stored in Out variable

Takes 20 or more minutes to run depending upon system compatibility

Patient similarity matrix creation : Integration Function in NetDx

If datatype n= 3

Three different PSN profiles need to created and stored in same directory

Each datatype generates multiple networks, and these are integrated into a single database by GeneMANIA

NetDx : Prediction runtime summary

```
--------------------------------
# patients = 525
# classes = 2 { LumA,notLumA }
Sample breakdown by class

   LumA notLumA
    230     295
2 train/test splits
Feature selection cutoff = 1 of 2
Datapoints:
        BRCA_mRNAArray-20160128: 17814 units
        clinical: 2 units


Custom function to generate input nets:
function(dataList, groupList, netDir,...) {
    netList <- c()
    # make RNA nets: group by pathway
    if (!is.null(groupList[["BRCA_mRNAArray-20160128"]])) {
    netList <- makePSN_NamedMatrix(dataList[["BRCA_mRNAArray-20160128"]],
```

```
------------------------------
         IS_TRAIN
STATUS    TRAIN TEST
  LumA      184   46
  notLumA   236   59
# values per feature (training)
        Group BRCA_mRNAArray-20160128: 17814 values
        Group clinical: 2 values
** Creating features
Pearson similarity chosen - enforcing min. 5 patients per net.
Made 3 RNA pathway nets
Made 2 clinical nets
Total of 5 nets
** Compiling features

** Running feature selection
        Class: LumA

   LumA nonpred    <NA>
    184     236       0
        Scoring features
        Writing queries:

                184 IDs; 2 queries (92 sampled, 92 test)
                Q1: 92 test;  92 query
```

Check the output files

- **We have two sub groups**

- **Need to check the prediction score of each class**

Check the output

```
> names(out)
[1] "inputNets" "Split1"    "Split2"
```

```
> out$Split1
```

Iteration 1 : ROC is 0.8

$accuracy
[1] 0.8

```
> out$Split2
```

$accuracy
[1] 0.7692308

Iteration 2 : ROC is 0.76

Check the Selected features

> out$Split1[1]

```
> out$Split1[1]
$featureScores
$featureScores$LumA
                                                          name score
1 GUANOSINE_NUCLEOTIDES__I_DE_NOVO__I__BIOSYNTHESIS.profile     2
2    MUCIN_CORE_1_AND_CORE_2__I_O__I_-GLYCOSYLATION.profile     2
3                              RETINOL_BIOSYNTHESIS.profile     1

$featureScores$notLumA
                                                          name score
1    MUCIN_CORE_1_AND_CORE_2__I_O__I_-GLYCOSYLATION.profile     2
2 GUANOSINE_NUCLEOTIDES__I_DE_NOVO__I__BIOSYNTHESIS.profile     2
3                                                 age_cont.txt     1
4                                               stage_cont.txt     1
5                              RETINOL_BIOSYNTHESIS.profile     1
```

There are three significant features in form of three different pathways crossed threshold criteria in LumA.

Check the Selected features

> out$Split1[2]

```
> out$Split1[1]
$featureScores
$featureScores$LumA
                                                          name score
1 GUANOSINE_NUCLEOTIDES__I_DE_NOVO__I__BIOSYNTHESIS.profile     2
2    MUCIN_CORE_1_AND_CORE_2__I_O__I_-GLYCOSYLATION.profile     2
3                                RETINOL_BIOSYNTHESIS.profile     1

$featureScores$notLumA
                                                          name score
1    MUCIN_CORE_1_AND_CORE_2__I_O__I_-GLYCOSYLATION.profile     2
2 GUANOSINE_NUCLEOTIDES__I_DE_NOVO__I__BIOSYNTHESIS.profile     2
3                                                 age_cont.txt     1
4                                               stage_cont.txt     1
5                                RETINOL_BIOSYNTHESIS.profile     1
```

There are three significant features in form of three different pathways crossed threshold criteria in LumA.

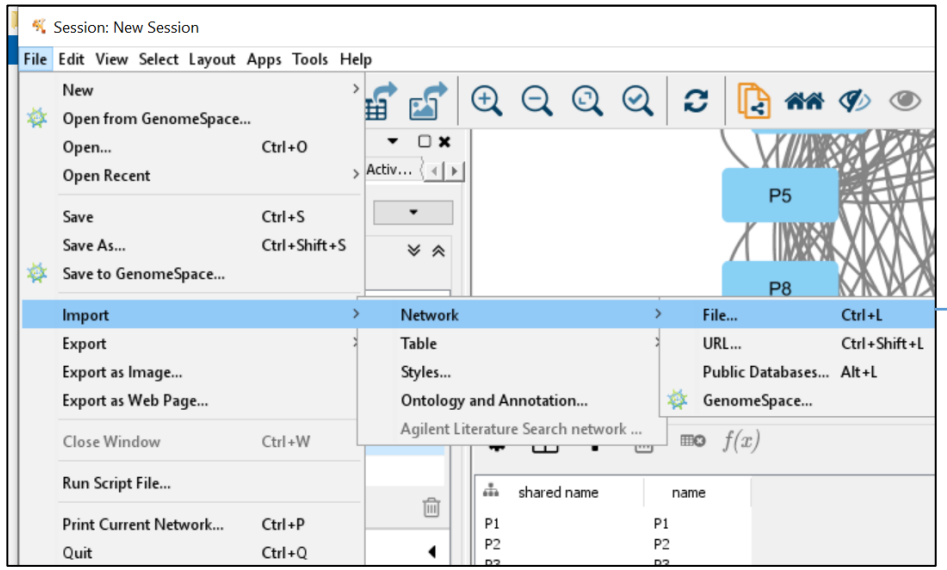
Similarity network of patients

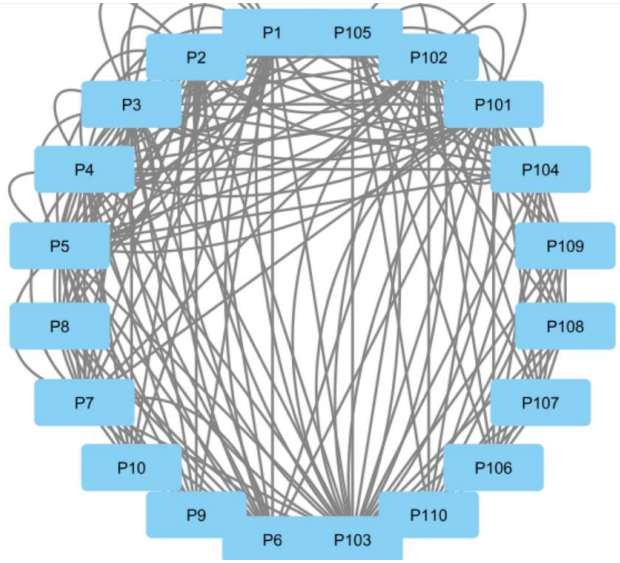
```
netDir <- sprintf("%s/extdata/example_nets",path.package("netDx"))
netFiles <- sprintf("%s/%s", netDir, dir(netDir,pattern="txt$"))
writeNetsSIF(netFiles,"merged.sif",netSfx=".txt")
```

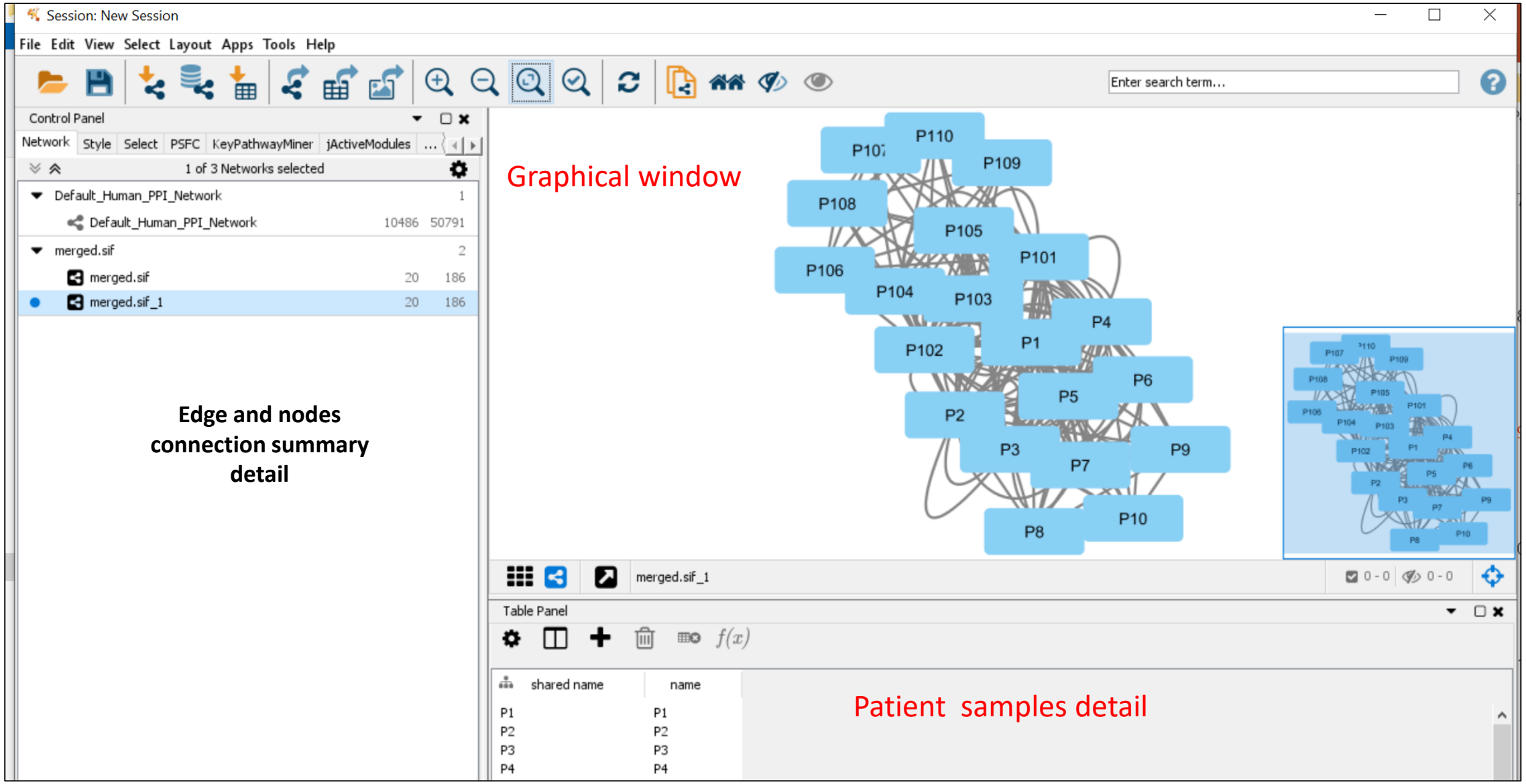
write patient networks in Cytoscape's .sif format

- **One can have plot in cytoscpae graphical windows as shown here**

Open cytoscape






Session: New Session
File Edit View Select Layout Apps Tools Help
Enter search term...
Control Panel
Network Style Select PSFC KeyPathwayMiner jActiveModules ...
1 of 3 Networks selected
Default_Human_PPI_Network 1
Default_Human_PPI_Network 10486 50791
merged.sif 2
merged.sif 20 186
merged.sif_1 20 186
Edge and nodes connection summary detail
Graphical window
P107
P110
P109
P108
P105
P101
P106
P104
P103
P4
P102
P1
P6
P5
P2
P3
P7
P9
P8
P10
merged.sif_1
0 - 0
0 - 0
Table Panel
shared name
name
P1 P1
P2 P2
P3 P3
P4 P4
Patient samples detail

GENE-GENE INTERACTION

GeneMANIA

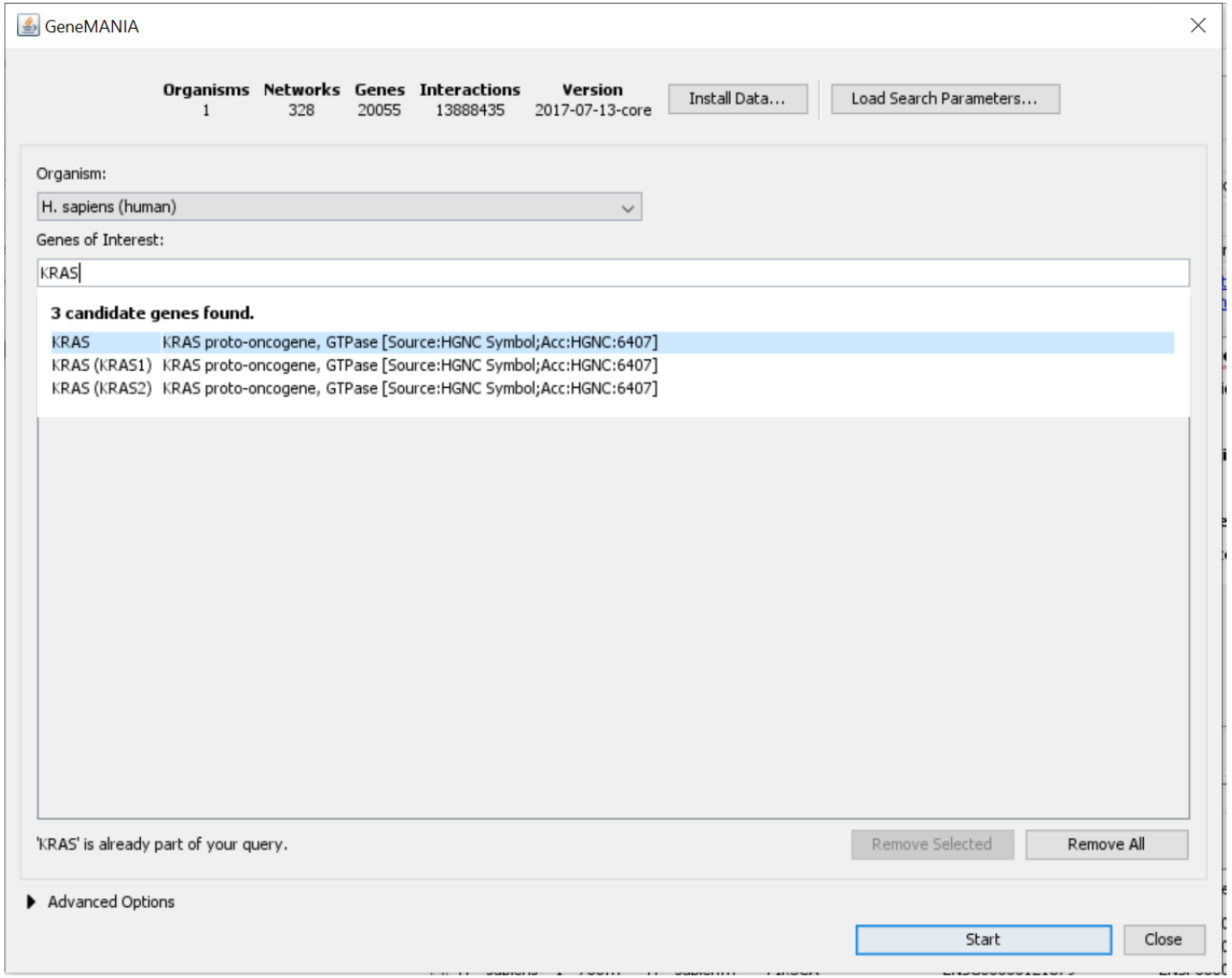
- **GeneMANIA uses a database of organism-specific weighted networks to construct the resulting composite network.**

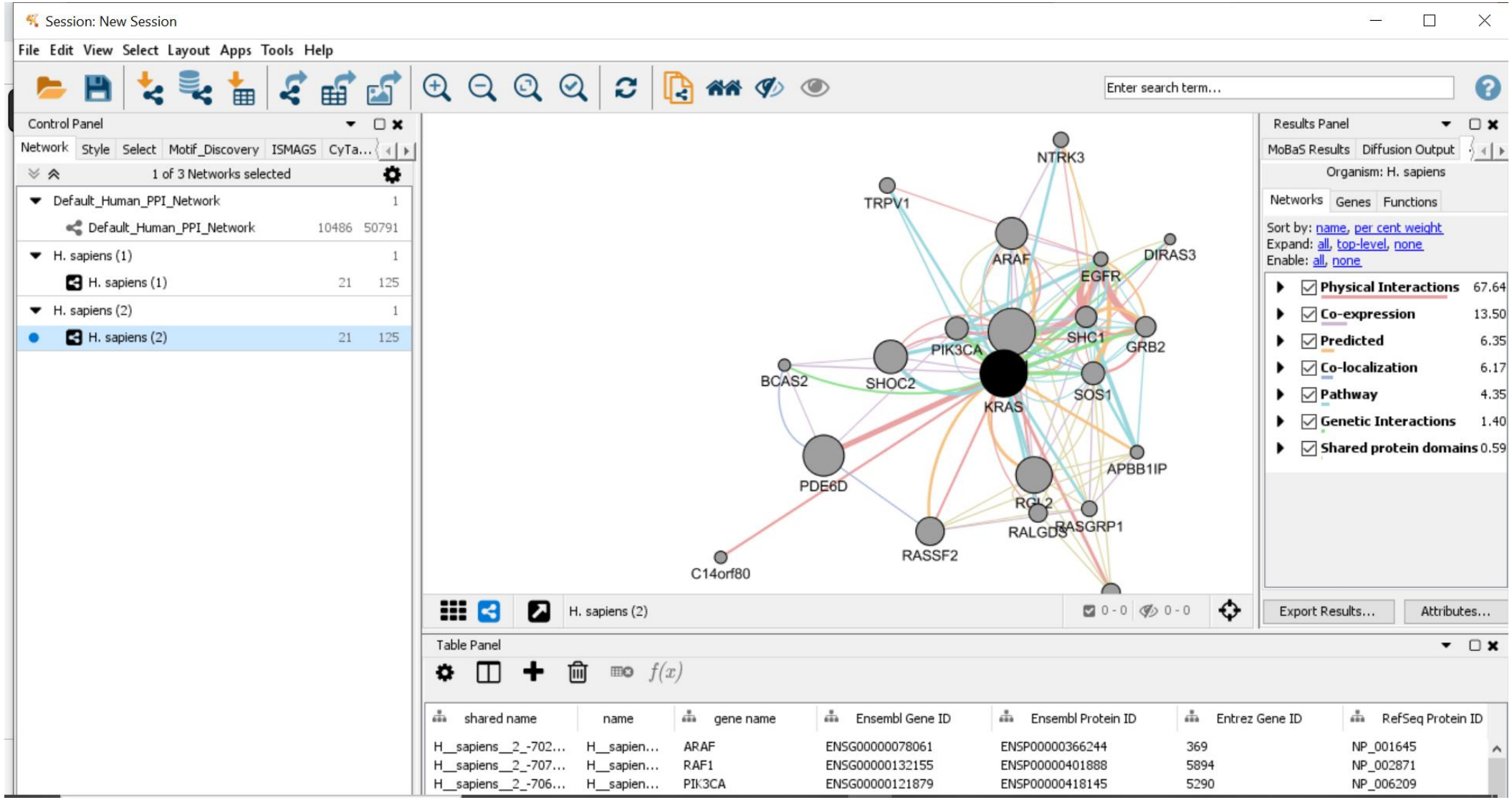
- **The database includes over 1800 networks, containing over 500 million interactions for 8 organisms: A. thaliana, C. elegans, D. melanogaster, D. rerio, H. sapiens, M. musculus, R. norvegicus, and S. cerevisiae.**

- **It could be used to predict the function of genes or gene sets.**

Let us use cytoscape Genemania APP

1. Open cytoscape
2. INSTALL GeneMANIA
3. Open GeneMANIA
4. In search bar, enter « KRAS »
5. ENTER START




Session: New Session
File Edit View Select Layout Apps Tools Help
Enter search term...
Control Panel
Network Style Select Motif_Discovery ISMAGS CyTa...
1 of 3 Networks selected
Default_Human_PPI_Network 1
Default_Human_PPI_Network 10486 50791
H. sapiens (1) 1
H. sapiens (1) 21 125
H. sapiens (2) 1
H. sapiens (2) 21 125
NTRK3
TRPV1
ARAF
DIRAS3
EGFR
SHC1
GRB2
PIK3CA
BCAS2
SHOC2
KRAS
SOS1
APBB1IP
PDE6D
RGL2
RALGDS
RASGRP1
RASSF2
C14orf80
H. sapiens (2)
0 - 0
0 - 0
Results Panel
MoBaS Results Diffusion Output
Organism: H. sapiens
Networks Genes Functions
Sort by: name, per cent weight
Expand: all, top-level, none
Enable: all, none
Physical Interactions 67.64
Co-expression 13.50
Predicted 6.35
Co-localization 6.17
Pathway 4.35
Genetic Interactions 1.40
Shared protein domains 0.59
Export Results...
Attributes...
Table Panel
shared name name gene name Ensembl Gene ID Ensembl Protein ID Entrez Gene ID RefSeq Protein ID
H__sapiens__2_-702... H__sapien... ARAF ENSG00000078061 ENSP00000366244 369 NP_001645
H__sapiens__2_-707... H__sapien... RAF1 ENSG00000132155 ENSP00000401888 5894 NP_002871
H__sapiens__2_-706... H__sapien... PIK3CA ENSG00000121879 ENSP00000418145 5290 NP_006209

Let us use Genemania to identify interaction of two query genes

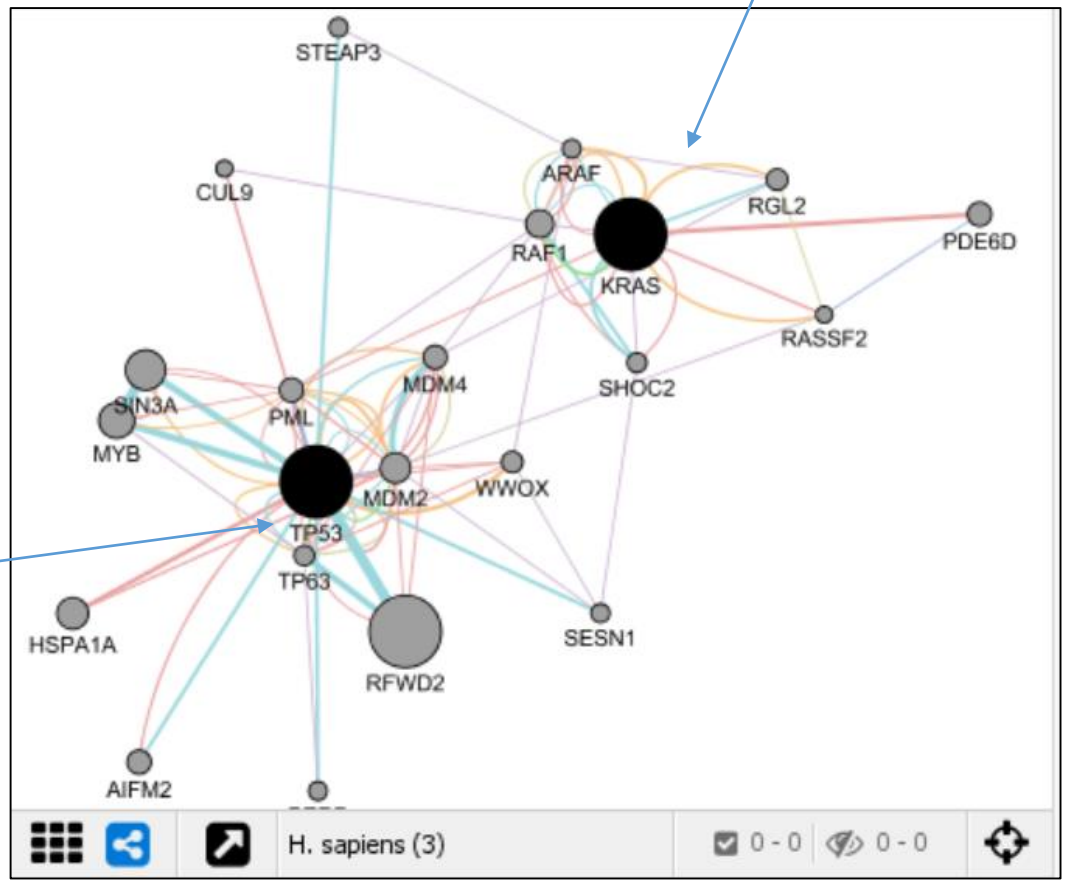
Gene A

1. In search bar, enter « KRAS » followed by TP53

Results indicates

- Some of the genes are in common regulation with these two genes.
- Both must be regulating same bioloigcal pathways.

Gene B



Exercise

- **Work with gene list :**

UGT1A10, UGT1A8, RPE, UGT1A7, UGT1A6, UGT2B28 , UGT1A5, CRYL1, UGDH, UGT2A1, GUSB, UGT1A9, DCXR

- **Identify their physical based interaction**
- **Identify their shared protein domains interaction**
- **Compare both the interaction graphs**