Artificial Intelligence in the Life Sciences 1 (2021) 100025

Contents lists available at ScienceDirect

Artificial Intelligence in the Life Sciences

journal homepage: www.elsevier.com/locate/ailsci

Methods & Protocols

OmicInt package: Exploring *omics* data and regulatory networks using integrative analyses and machine learning

Auste Kanapeckaite[a,b]

[a] *Algorithm379, Laisvės g. 7, Vilnius, Lithuania*
[b] *University of Reading, School of Pharmacy, Hopkins Building, Reading RG6 6UB United Kingdom*

ARTICLE INFO

Keywords:
Machine learning
Gaussian mixture models
Omics analyses
Epigenomics
Gene networks

ABSTRACT

OmicInt is an R software package developed for a user-friendly and in-depth exploration of significantly changed genes, gene expression patterns, and the associated epigenetic features as well as the related miRNA environment. In addition, *OmicInt* offers single cell RNA-seq and proteomics data integration to elucidate specific expression profiles. To achieve this, *OmicInt* builds on a novel scoring function capturing expression and pathology associations. The developed scoring function together with the implemented Gaussian mixture modelling pipline helps to explore genes and the linked interactome networks. The machine learning pipeline was designed to make the analyses straightforward for the non-experts so that researchers could take advantage of advanced analytics for their data evaluation. Additional functionalities, such as protein type and cellular location classification, provide useful assessments of the key interactors. The introduced package can aid in studying specific gene networks, understanding cellular perturbation events, and exploring interactions that might not be easily detectable otherwise. Thus, this robust set of bioinformatics tools can be very beneficial in drug discovery and target evaluation. *OmicInt* is designed to be freely accessible to involve a larger bioinformatics community and continuously improve the developed algorithmic methods.

1. Introduction

OmicInt is an R software package developed for an in-depth exploration of significantly changed genes, gene expression patterns, and the associated epigenetic features as well as the related miRNA environment. The package helps to assess gene clusters based on their known interactors (proteome level) using several different resources, e.g., UniProt and STRING DB [1–3]. Moreover, *OmicInt* provides an easy Gaussian mixture modelling [4–6] pipeline for an integrative analysis that can be used by a non-expert to explore gene expression data. Specifically, the package builds on a previously developed method to explore gene networks using significantly changed genes, their log-fold-change values (LFC), and the predicted interactome complexity [5]. This approach can aid in studying specific gene networks, understanding cellular perturbation events, and exploring interactions that might not be easily detectable otherwise [5]. To this end, the package offers many different utilities to help researchers quickly explore their data in a user-friendly way where machine learning is made easily accessible to non-experts (Figs. 1 and 2). It is also important to highlight that the lack of freely available tools to explore complex expressome data motivated the creation of this set of tools. For example, commercial solutions, such as Clarivate analytics [7], are almost inaccessible to individual users because of the very expensive software. Freely available tools, namely GeneMANIA or Cytoscape platforms [8–11], while very useful, do not permit machine learning applications or complex regulome integration. Thus, seeing the existing need for *omics* dedicated tools that could evolve as more bioinformaticians get involved encouraged creating the *OmicInt* package.

Machine learning which offer effective methods to assess multidimensional biological data is also a very important part of the developed package. For the purpose of biological data evaluation, Gaussian mixture models (GMMs) were selected as they employ a probability based classification where each data point assignment has a different probability of belonging to one of the clusters [4–6]. The probabilistic nature of GMM relies on the assumption that the data can be explained by a finite mixture of Gaussian distributions with unknown parameters [4]. As a result, this is a soft classification method that is more suitable to assess biological parameters in comparison to hard classification techniques (e.g., k-means) [4–6]. This is because gene or protein interaction networks are dynamic systems and probabilistic feature separation allows for more flexibility in defining boundaries between groups [5]. Moreover, the extracted probability values can be incorporated into other analytical pipelines to further refine the data. The developed GMM pipeline automates the assessment of the information criterion to optimise the number of clusters for modelling and also predicts the best suited model for the expectation-maximisation (EM) algorithm which

E-mail address: auste.kan@algorithm379.com

https://doi.org/10.1016/j.ailsci.2021.100025
Received 25 November 2021; Accepted 26 November 2021
Available online 11 December 2021

Example table with only LFC values

	A	B	C
1	Symbol	log2FoldChange	pvalue
2	SAR1A	-2.18777269502012	2.65652091646361E-05
3	C6orf62	-2.6742131704395	1.6915673694844E-07
4	AXL	-2.78650785919511	0.000173959539412
5	BICC1	-3.59855326113771	0.00027388866015
6	CAPZA1	-1.73278403064529	0.000232183462116
7	TXNIP	1.46062912017625	7.34720482186151E-05
8	HNRNPH1	-1.81995372081358	0.000592319156385
9	RAB31	-1.79837938364367	0.00041973140141
10	UBE2B	-2.06138280667398	0.000125275768063
11	PAFAH1B2	-1.56007182816026	0.001391919840109
12	EIF2S3	-1.74298250448705	0.001063319998638
13	YWHAG	-1.55076880524874	0.000815638927099
14	ENAH	-1.69808760167615	0.000269611887499
15	PPP3CA	-2.67216823748962	3.98459567587323E-06

Example table with LFC, beta, and gamma values

	A	B	C	D	E
1	Symbol	log2FoldChange	pvalue	beta	gamma
2	SAR1A	-2.18777269502012	2.65652091646361E-05	0.25	0
3	C6orf62	-2.6742131704395	1.6915673694844E-07	0	0.3
4	AXL	-2.78650785919511	0.000173959539412	0	0.56
5	BICC1	-3.59855326113771	0.00027388866015	0	0
6	CAPZA1	-1.73278403064529	0.000232183462116	0.4	0
7	TXNIP	1.46062912017625	7.34720482186151E-05	0	0.75
8	HNRNPH1	-1.81995372081358	0.000592319156385	0	0
9	RAB31	-1.79837938364367	0.00041973140141	0	0.02
10	UBE2B	-2.06138280667398	0.000125275768063	0.41	0
11	PAFAH1B2	-1.56007182816026	0.001391919840109	0	0
12	EIF2S3	-1.74298250448705	0.001063319998638	0.7	0
13	YWHAG	-1.55076880524874	0.000815638927099	0.8	0
14	ENAH	-1.69808760167615	0.000269611887499	0	0.015
15	PPP3CA	-2.67216823748962	3.98459567587323E-06	0	0

Metadata file example

	A	B
1	Sample_ID	Condition
2	CAD1	hypertension
3	CAD2	hypertension
4	CAD3	hypertension
5	CAD4	hypertension
6	CAD5	hypertension
7	CAD6	hypertension
8	CAD10	hypertension
9	CAD11	hypertension
10	N10	healthy
11	N12	healthy
12	N13	healthy
13	N14	healthy
14	N15	healthy
15	RF2	CKD

Normalised count table example

	A	B	C	D	E	F
1	Symbol	CAD1	CAD2	CAD3	CAD4	CAD5
2	MT-CYB	9384.55930132127	11923.5039503911	34985.4747081763	4216.00298751307	12402.4413183367
3	MT-ND4	10934.2964538035	12360.328040959	24509.2381415289	5360.27179089838	12871.5303032926
4	MT-CO1	9722.28571644071	11516.7510834493	12963.1580920346	5587.18655703799	11594.0474618033
5	MT-CO3	8205.9264136993	10957.0042717434	17443.7581673424	5740.20990972336	13001.314416671
6	FN1	192.371263022342	249.538399563515	211.313998100786	748.609107229636	466.544198419069
7	COL1A2	138.536021504896	178.317080449193	135.124152051885	504.138579737424	331.670511967011
8	ACTB	777.02198590181	690.05544741877	1249.03562947764	1889.20954257112	2393.7958689793
9	MT-ATP6	6347.89277812717	7713.00507741775	11965.5224075336	3373.85049516584	7503.72723493652
10	MT-CO2	5702.58768313804	4504.35275998582	8569.36665664999	1882.92091163884	5782.60327385963
11	MT-ND1	6299.08215915135	9499.34127387032	10046.9718242743	4573.4068454974	11624.5849002453
12	MT-ND5	3404.89957517343	3910.3142020397	5409.21359962516	2197.61448454146	2744.12814888307
13	MALAT1	6320.61625575833	6583.48771279727	2876.10032088432	2196.56637938608	3348.09082029103
14	SAR1A	3043.84455539642	4617.25173991519	9943.43858399882	3869.86625994907	4469.49342085468
15	MT-ND4L	2144.79602205507	3601.68848587763	3118.47429106776	1752.16979350528	3407.46917281709

Fig. 1. Examples for the required data formats which include the normalised gene expression values, log fold change (LFC) values, and the meta data file.

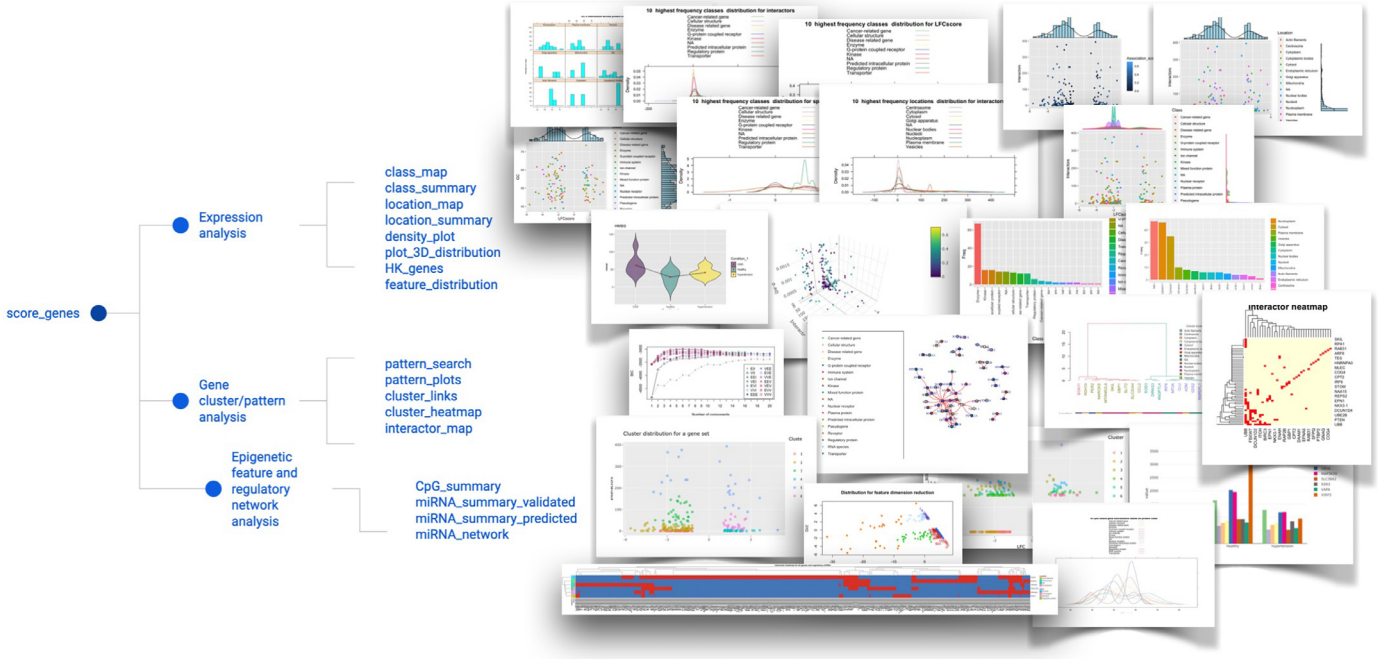

Fig. 2. Schematic representation of package functions and specific analyses.

helps to maximise the likelihood of data point assignments [4,12]. As a result, the users do not need to have an extensive knowledge to fine-tune their GMM parameters as the process is streamlined for them.

The key analytical parameter in the machine learning pipeline and exploratory analyses is a specific score, namely LFC_{score}, which can have a different derivation depending on the selected parameters Eqs. (1)–(3). The user has several options to select from since the equations were expanded with additional data based on the earlier derivation of the multi-*omics* Eq. (5). The score α values are downloaded automatically from curated database images which were generated via text mining to retrieve, update, and integrate data in an easier-to-use format (i.e., database image) for the analyses. Databases used include Disgenet,

Uniprot, and STRING DB [1,3,13]. For example, α_{asoc} score allows to infer how strongly a gene is linked to a disease or pathological phenotype ranging from 0 (no link) to 1 (the strongest association) Eq. (1) [13]. Similarly, α_{spec} captures how specific a gene is when describing the pathology Eq. (2) [13]. Association scores are based on different curated resources as described earlier [13]. The user can choose from different types of scores ("association_score", "specificity_score", or the geometric mean of both) when selecting the type of the equation for LFC_{score}. Scores βcell and γprot are the scaled values for single cell and proteome data, respectively. That is, βcell has to be provided by the user if they have such experimental information integrated where a gene value from a single cell data cluster is extracted using a pseudo-bulk differential gene expression approach. The LFC scores from pseudo-bulk data need to be scaled according to the Eq. (4). The same approach should be applied when calculating γprot for protein (corresponding gene) values.

$$LFC_{score} = LFC\left(1 + \alpha_{asoc} + \beta_{cell} + \gamma_{prot}\right) \quad (1)$$

LFC_{score} equation where LFC - Log Fold Change, base 2; α_{asoc} - a disease association score; β_{cell} - scaled single cell LFC; γ_{prot} - scaled proteome LFC.

$$LFC_{score} = LFC\left(1 + \alpha_{spec} + \beta_{cell} + \gamma_{prot}\right) \quad (2)$$

LFC_{score} equation where LFC - Log Fold Change, base 2; α_{spec} - a disease specificity score; β_{cell} - scaled single cell LFC; γ_{prot} - scaled proteome LFC.

$$LFC_{score} = LFC\left(1 + \sqrt{(\alpha_{soc}\ \alpha_{spec})} + \beta_{cell} + \gamma_{prot}\right) \quad (3)$$

LFC_{score} equation where LFC - Log Fold Change, base 2; α_{asoc} and α_{spec} are integrated using a geometric average score; β_{cell} - scaled single cell LFC; γ_{prot} - scaled proteome LFC.

$$LFC_{scaled} = LFC_{gene}/LFC_{median} \quad (4)$$

β_{cell} or γ_{prot} scaling example where LFC_{gene} - a gene specific value and LFC_{median} - a median value for all available LFC values per specific condition and gene set.

OmicInt provides many other valuable tools to map the interactome using information on the target cellular location or protein class/function type. In addition, density functions allow for an exhaustive assessment of gene distributions which may hint at potential functions or dominant processes within a specific condition. Epigenetic feature (CpG islands, GC%) and miRNA exploration tools also provide additional information on the epigenome and non-coding regulome which might be relevant for some genes and conditions, especially if a higher enrichment of these patterns can be found. Currently, the analyses are only available for human data sets. The software package is freely distributed via Github and CRAN repositories to make the analyses accessible to researchers [14,15]. Github environment also provides opportunities to submit requests or suggestions and participate in further algorithm development [14].

2. Methods

OmicInt package architecture (Fig. 2) is divided into gene expression, gene cluster/pattern, and epigenetic feature/regulatory network analysis with a detailed vignette to guide the user [14,15]. Machine learning pipeline is based on Gaussian mixture models which is designed to include the optimal cluster number (Bayesian information criterion), automatic model fitting during the expectation maximisation phase of clustering, model-based hierarchical clustering, as well as density estimation and discriminant analysis [4,12]. The package enables advanced options to perform a user-specified clustering to use the data in other workflows. *OmicInt* also retrieves data from multiple databases by generating combined and curated database images for easier use [1,3,13]. The package was built using functional programming principles and the analyses were benchmarked using the following studies distributed via NCBI GEO database [16]: GSE160145, GSE3585, GSE26887, and GSE116250.

3. Results

3.1. Data preprocessing

Before starting the analysis the user must ensure that the supplied data is in the right format. There are several different options to prepare a data frame (CSV format) that contains all the relevant experimental information Fig. 1; Eqs. (1)–(4). Depending on the selection, the downstream analyses will provide interactive graphs and maps (Fig. 2). Consistent data preparation and integration allow for a stable processing workflow which enables an efficient organisation of data sets.

Data pre-processing relies on the *score_genes* function that collects data from the STRING database and other disease association data sets to scale and prepare additional score integration [3,13]. Several key parameters should be provided; the *data* parameter requires a data frame containing gene names as row names and a column with LFC values. The example is provided in Fig. 1; the parameter *alpha (α)* has a default value set as "association" which gives a score from 0 to 1 based on how strongly a gene is associated with a pathological phenotype; other options are "specificity" - to give values based on how specific a gene is when describing a disease and "geometric" - to give a geometric mean score of both association and specificity. The α score is calculated automatically for the genes in the data set. In addition, it is possible to add weighted single cell and proteomics data by selecting additional parameters. The parameter *beta* is set to have a default value as FALSE; if TRUE, the user needs to supply data with a column *beta* that contains information on gene associations from single cell studies. Similarly, parameter *gamma* has a default value FALSE; if TRUE, the user is required to supply data with a column *gamma* that contains information on gene associations from proteome studies. The function returns a data frame for the downstream analyses.

```
#Code example for data preprocessing
#data<-score_genes("data.csv")
#head(data)

#   Symbol log2FoldChange       pvalue Interactors Association_score
#1   SAR1A      -2.187773 2.656521e-05          24         0.0000000
#2 C6orf62      -2.674213 1.691567e-07           0         0.0000000
#3     AXL      -2.786508 1.739595e-04           2         0.3230769
#4   BICC1      -3.598553 2.738887e-04           3         0.3000000
#5  CAPZA1      -1.732784 2.321835e-04          66         0.3789474
#6   TXNIP       1.460629 7.347205e-05          30         0.3000000
#  Specificity_score  LFCscore
#1             0.000 -2.187773
#2             0.000 -2.674213
#3             0.590 -3.686764
#4             0.751 -4.678119
#5             0.601 -2.389418
#6             0.631  1.898818
```

3.2. Exploratory analyses

Function *density_plot* plots a density plot for gene expression data prepared by the *score_genes* function. The plots can be used for a quick assessment and summarisation of the overall parameters (Fig. 3). Specifically, the plots allow the evaluation of how key parameters, such as LFC, LFC_{score}, and disease association or specificity scores, associate with the highest frequency protein classes and cellular locations. For example, the most frequent protein classes may have specific distribution patterns hinting at predominant cellular processes. Similarly, examining distributions for cellular locations might highlight the most involved and/or affected cellular strictures.

```
#An example of a function call to get density plots
#density_plot(data)
```

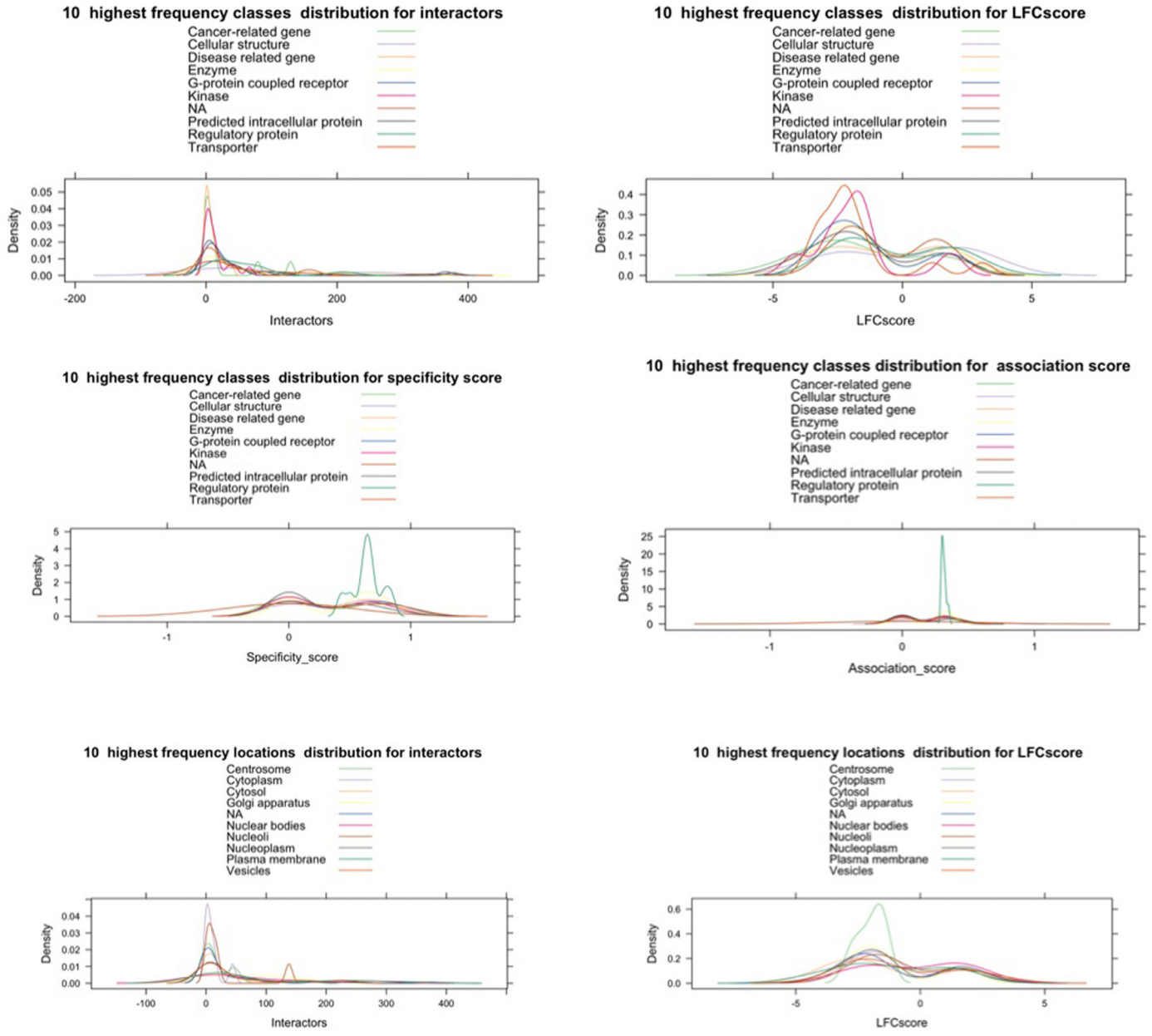

Fig. 3. Density plot examples for different parameters.

Function *feature_distribution* also provides a way to visualise main feature distributions through density plots combined with LFC_{score} and interactor number scatter plots (Fig. 4). These plots allow to quickly assess if there are any dependencies between LFC_{score} and the interactor numbers. Such plots also help to see if any obvious gene clusters emerge. In early analyses this can aid in understanding whether the expression is dependent on any cellular site or protein class which could suggest a specific functional enrichment. This function might issue a warning if the data points were missing or too few for density plotting; however, it does not affect the overall visualisation.

```
#A simple call to implement the feature distribution analysis
#feature_distribution(data)
```

Function *plot_3D_distribution* allows to explore 3D distributions between the number of interactors, LFC_{score}, and p.adj values. In addition to providing a data parameter, the user can select how to color data points depending on the association or specificity score (e.g., selecting "specificity") (Fig. 5). This analysis can help identify specific clusters for the expression patterns and interactors based on the significance of how the gene expression changed in a given condition. In addition, data point coloring based on gene association or specificity in the context of diseases can help capture additional patterns in the data.

```
#A function call example to explore the data in the interactive 3D plot
#plot_3D_distribution(data)
```

Function *class_summary* provides analysis on main protein classes where a barplot helps to visualise the class distribution. Similarly, the function *location_summary* summarises the location distribution data (Fig. 6). Assessing this information can highlight if there are any specific biases in data for target location or function which might indicate underlying cellular perturbations or changes in the function.

```
#Functions class_summary and location_summary are called for the preprocessed data
frame

#class_summary(data)

#Location_summary(data)
```

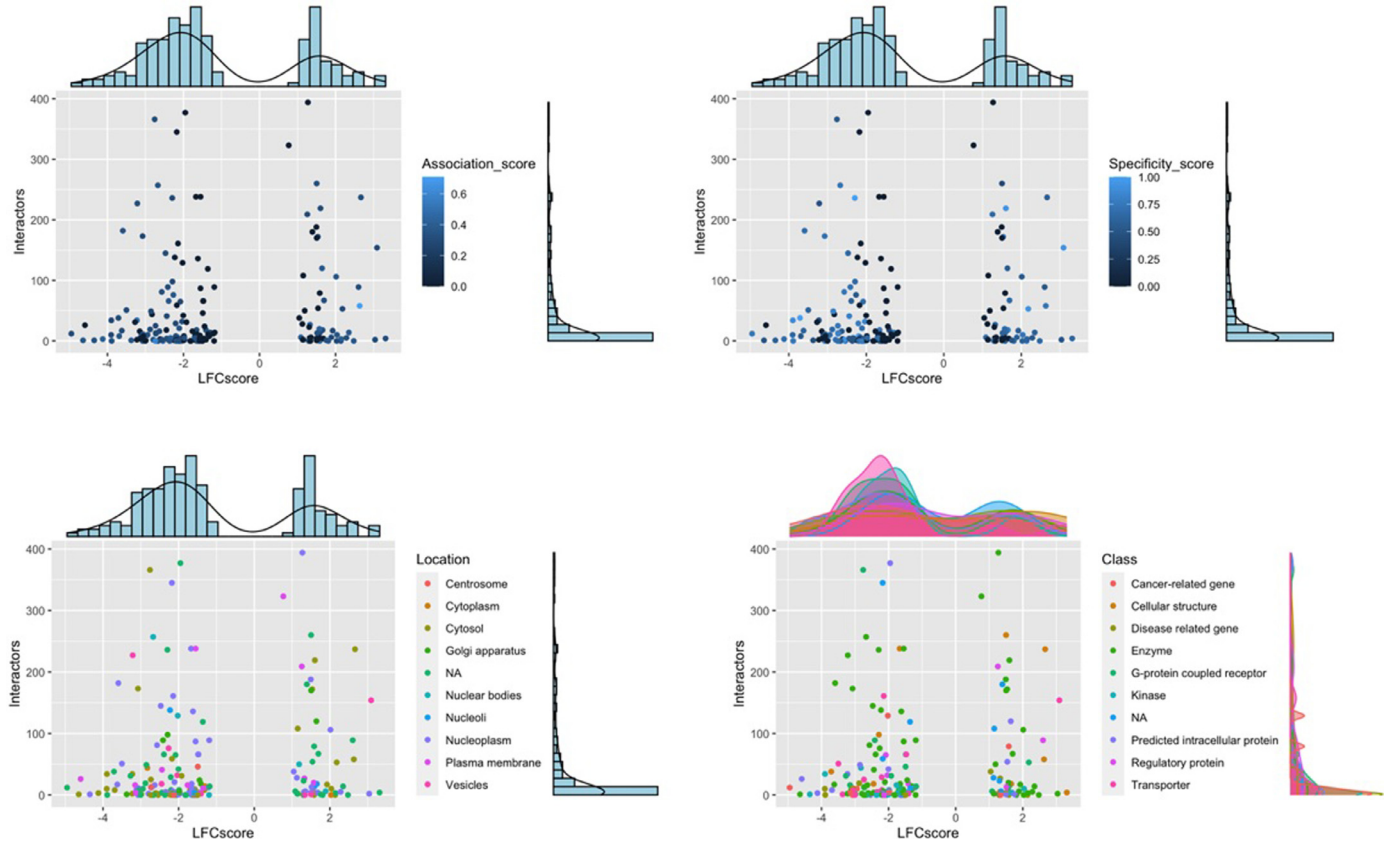

Fig. 4. Feature distribution plot examples.

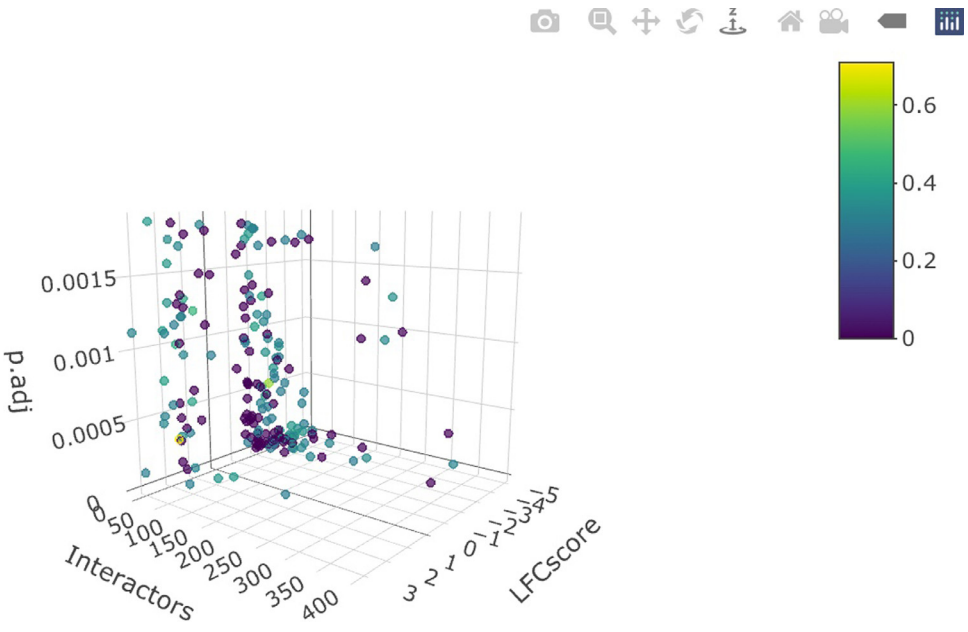

Fig. 5. Interactive 3D feature distribution.

Function *location_map* allows the visualisation of how the highest and lowest LFC_{score} genes cluster based on the protein cellular location data (Fig. 7). The user can specify the number of the top and lowest genes to consider. The function returns a dendogram generated based on LFC_{score} values. The "euclidean" method is used for distance calculation and the "Ward.D2" method - for hclust generation. Gene labels are colored to indicate major clusters where the hclust generated cluster number is doubled to select for more subgroups. In addition, to achieve a finer separation of lower dendogram branches the following equation is used to set the height for the color differentiation of different branches Eq. (5).

This equation takes the mean value for hclust function height calculation and multiplies by the dendrogram cluster number scaled twice. The plot also provides cellular location visualisation for each gene (Fig. 7).

$$H_{dendogram} = \left(hclust_{height}/hclust_{n}\right) \cdot dendogram_{cluster\_number} \cdot 2 \quad (5)$$

The height calculation for the color differentiation of different dendogram branches.

```
#Location mapping is done using a single function call
#Location_map(data)
```

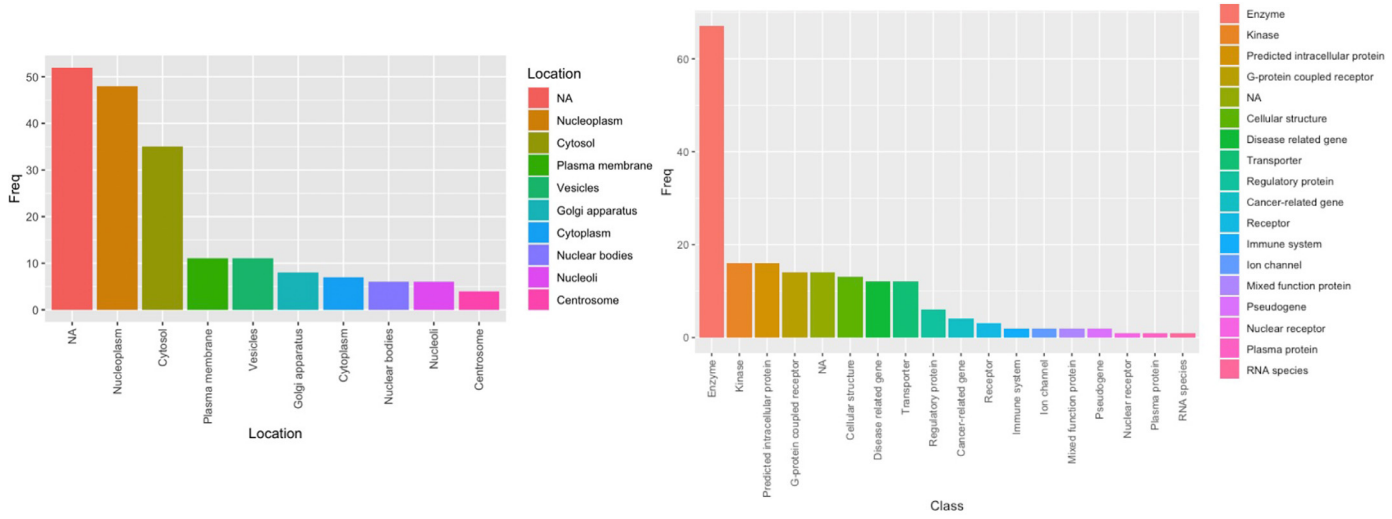

Fig. 6. Location and class summary plots; NA – no classification available.

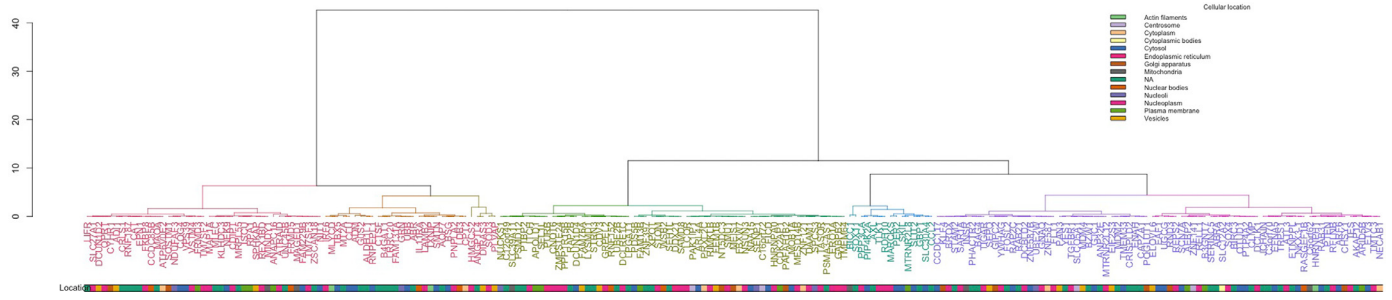

Fig. 7. Dendogram with mapped cellular locations where coloured gene symbols represent the identified clusters and coloured branches show smaller subclusters. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Similarly, the function *class_map* provides a visualisation of how the highest and lowest LFC_{score} genes cluster based on protein class. In addition to a data frame generated by *score_genes*, the function also requires a *num* parameter to specify the number of genes to consider from the top upregulated and downregulated genes, if this option is not selected all genes will be used (Fig. 8).

```
#Class mapping is done using a single function call
#class_map(data, 20)
```

HK_genes function provides a convenient overview of the housekeeping genes and allows to check if these genes varied throughout conditions. Depending on the number of conditions separate plots will be generated (Fig. 9). Inspecting housekeeping genes can help understand if there was any significant variation between sample groups which might have arisen from biological or technical variation.

```
# To retrieve housekeeping gene assessment only a single function call is required
#HK_genes(data)
```

3.3. Gene cluster and expression pattern analyses

Function *cluster_genes* helps to select an optimal number of clusters and a model to be fitted during the EM phase of clustering for GMM. The function provides summaries and helps to visualise gene clusters based on generated data using *score_genes* function. Weighted gene expression is clustered based on the interactome complexity, i.e., the number of known interactors according to the STRING database [3], with a cut-off of 700 for the score threshold. The threshold is set automatically to control for the reliability of the interactions [2,3]. The function also provides scatter and dimension reduction plots to analyse the clusters and features in the data (Fig. 10). Required parameters include a data frame containing a processed expression file from *score_genes* with LFC_{score} and a *max_range* number for cluster exploration during the model selection (the default value is 20 clusters). The *clusters* parameter can be provided for the number of clusters to test when the cluster number estimation is not based on the best BIC output (the user then also needs to supply *modelNames*). This option allows users to perform GMM for a specific number of clusters. The *modelNames* parameter can only be supplied when the *clusters* value is also specified. This option will model the data based on the user parameters for the cluster assignment (Fig. 10) which can be helpful if a different number of clusters helps to explain the data better. The function not only provides a summarised modelling output and plots but also returns a data frame with assigned clusters which can be used by more advanced users in other machine learning pipelines or data comparison studies. For example, gene set clustering based on the interactome size provides insights on the emerging patterns for gene expression changes and the size of the involved network. Selecting specific genes can help build signalling networks based on the identified seed points. Feature distribution analysis also helps to assess the emerging trends in the data based on the variability. That is, gene variation patterns in the experiment might indicate functionally related groups which could be used to reconstruct relevant pathways (Fig. 10). The user is advised to set seed before using the function to get reproducible results.

```
#A machine learning pipeline is implemented using a simple call
#The output data frame can be stored in a new object for further analyses
#model_report<-cluster_genes(data)
```

```
# The function will automatically output the Bayesian information criterion (BIC) and
the type of the model for fitting
# Detailed explanation of the model selection is provided with a dependency package –
Mclust.


#Best BIC values:
#             VVI,6        VVI,7        VVI,5
#BIC      -2481.986 -2483.544400 -2485.429861
#BIC diff     0.000    -1.558131    -3.443592
# An example of the model report summary output
# head(model_report)

#       Interactors  LFCscore Cluster Symbol
#CAPZA1          66 -2.389418       1 CAPZA1
#RAB31            0 -2.542398       2  RAB31
#UBE2B            2 -2.061383       2  UBE2B
#YWHAG           21 -2.111448       1  YWHAG
#ENAH            29 -2.207514       1   ENAH
#PPP3CA          51 -3.500322       1 PPP3CA
```

Function *cluster_links* provides the same Gaussian mixture modeling pipeline as *cluster_genes*; however, instead of the interactor number clustering, the user can select a specific disease score *type* (the default selection is "association"). This parameter can define either the association or specificity for a disease, i.e., if the gene has known links to disease phenotypes and how specific it is when describing a pathology. The function also provides scatter and dimension reduction plots to analyse the clusters and features in the data. An additional output is a model report summarising the cluster assignments which can be used in other modelling analyses. This information can be used to compare association and network size influences for different clusters and gene expression patterns.

Function *pattern_search* explores the occurrences of specific patterns in gene sets. That is, it searches each condition for emerging patterns (e.g., if multiple conditions are provided) to group genes that changed in a similar manner (Figs. 11 and 12). The search algorithm works by first generating potential patterns to search depending on the number of subclasses. For example, if a condition has several subclasses as in the case example, where Condition 1 has healthy, hypertensive, and chronic kidney disease (CKD) groups, then potential pattern scenarios are generated, e.g., "up-up-up" or "down-up-down". Following this, the overall expression for each gene is calculated using geometric mean across all conditions, this gives a basal line against which an individual gene expression value is weighed to deduce if it is in a 'up' or 'down' state. Comparing against a baseline is a more universal approach than performing a pair wise comparisons which may not be effective for multiple subclasses or complex interactions. In addition, averaging expression using a geometric mean method provides a baseline for comparisons taking into account all the extreme values which might result either from biological or technical effects. It is important to note that taking a geometric mean might not be optimal in all cases, but in a balanced experiment it should provide additional information for the downstream analyses. The function returns a summary of how many genes are identified for each pattern type across conditions.

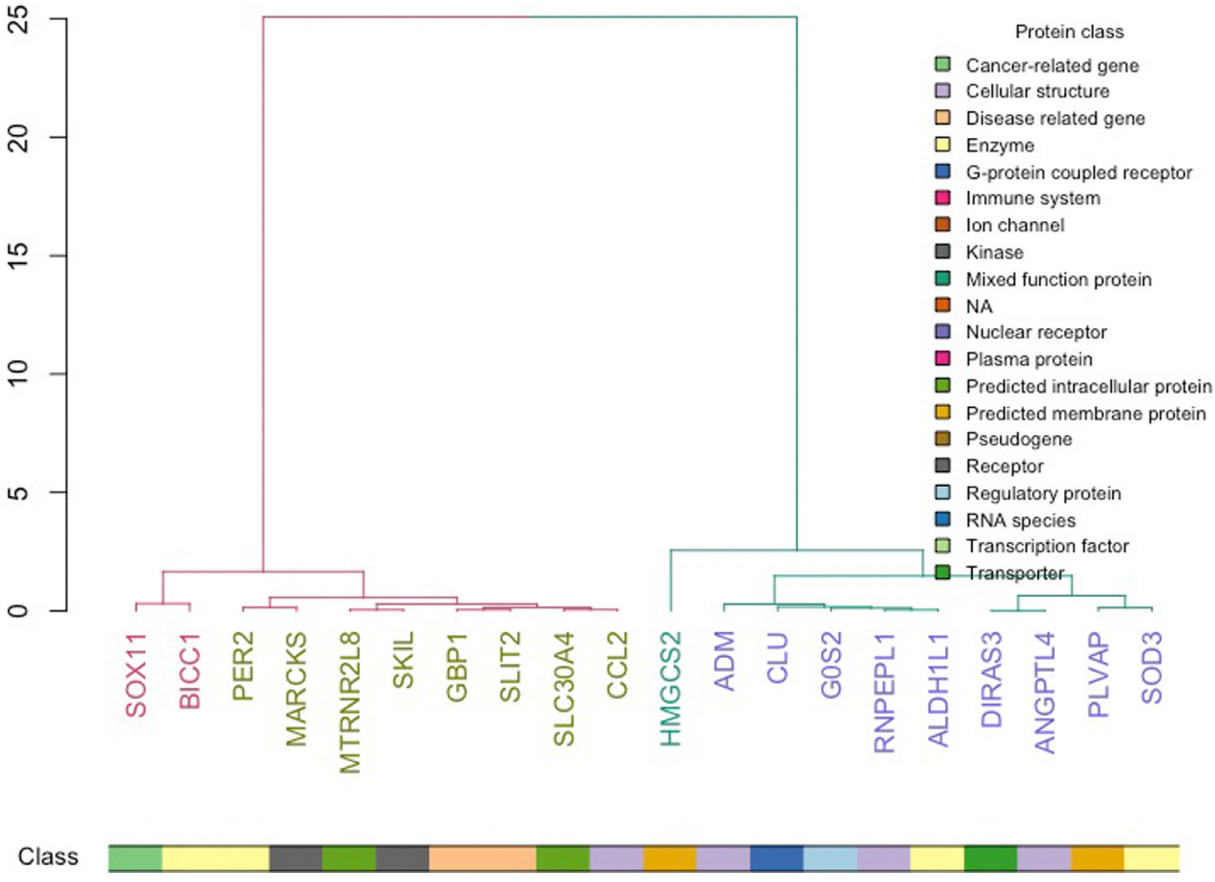

Fig. 8. Dendogram with mapped protein functions where coloured gene symbols represent the identified clusters and coloured branches show smaller subclusters. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

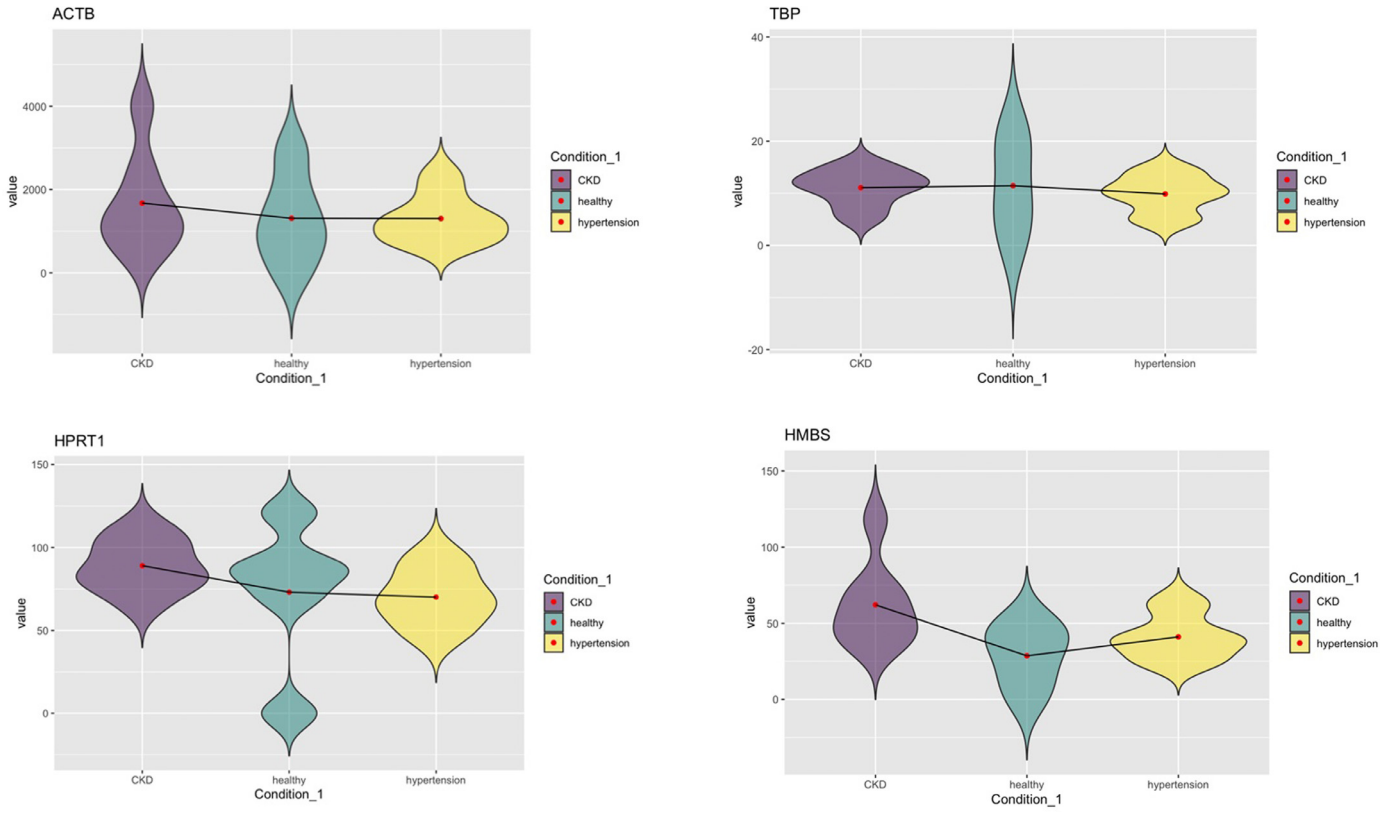

Fig. 9. An example of the housekeeping gene distribution. The red marker indicates the mean for the group and violin plots allow to assess global distribution patterns. CKD – chronic kidney disease patient group, healthy – healthy population group, and hypertension – hypertensive patient group. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

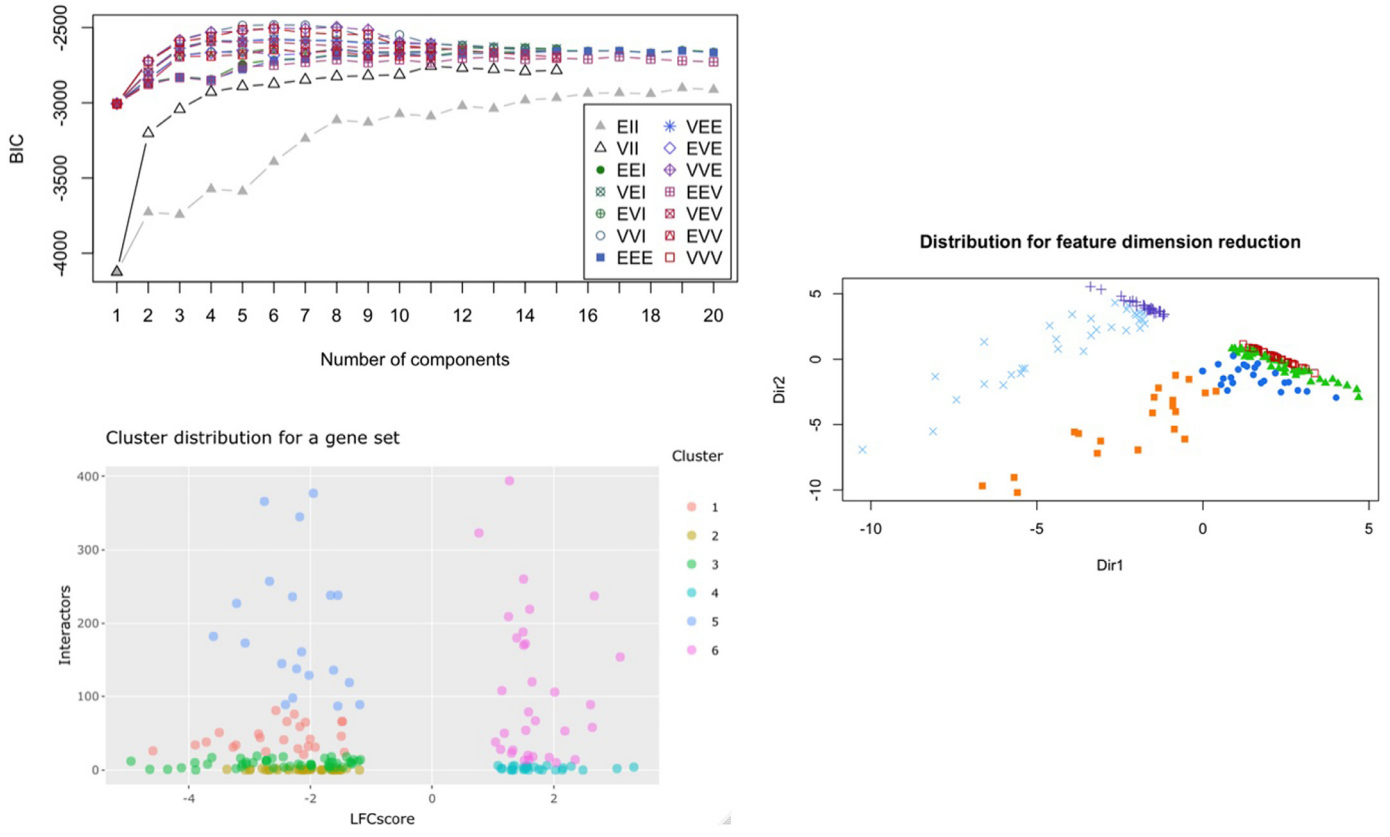

Fig. 10. GMM analysis examples showing the Bayesian information criterion (BIC) evaluation and model type prediction, clustering analysis, as well as the dimension reduction analysis based on the intrinsic variability within the data.

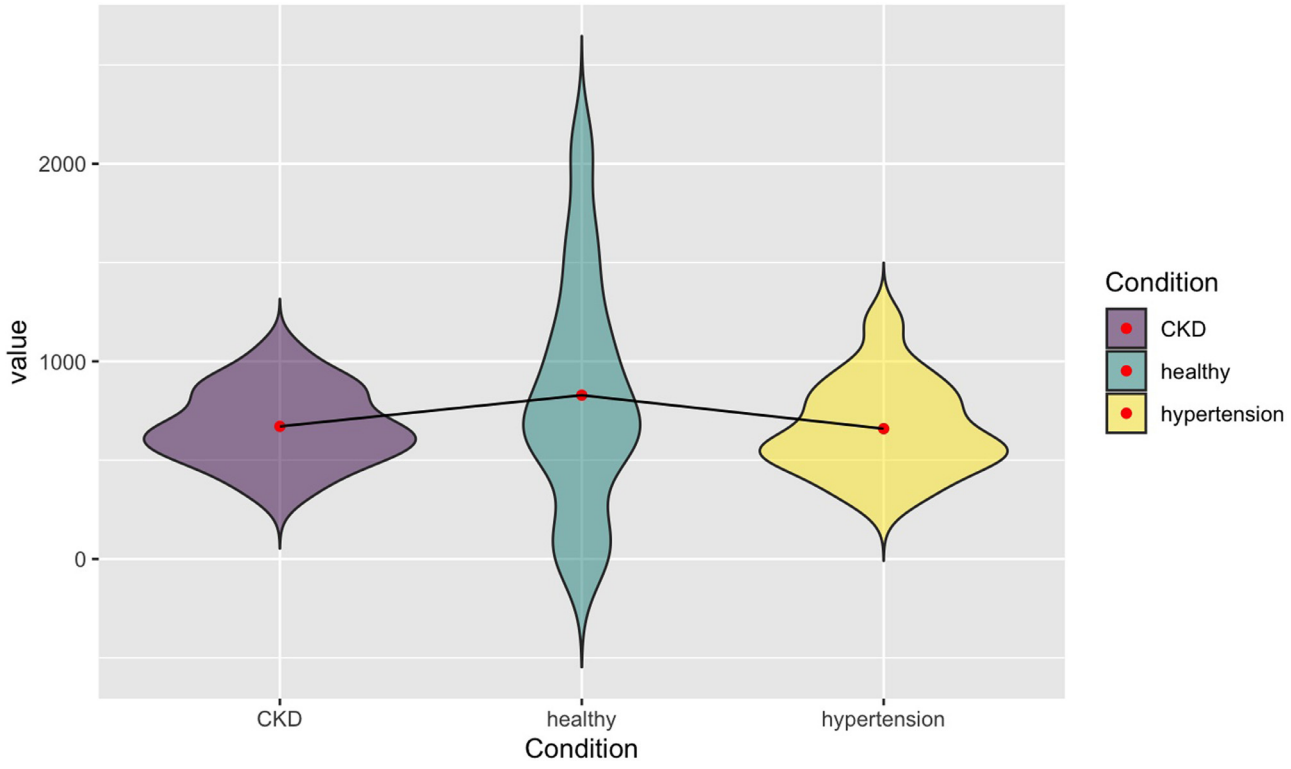

Fig. 11. Gene distribution patterns for a specific expression pattern subset where mean values (signified with a red point) are connected to highlight the pattern features with respect to the mean value (the example is from a "down-up-down" pattern group). CKD – chronic kidney disease patient group, healthy – healthy population group, and hypertension – hypertensive patient group. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

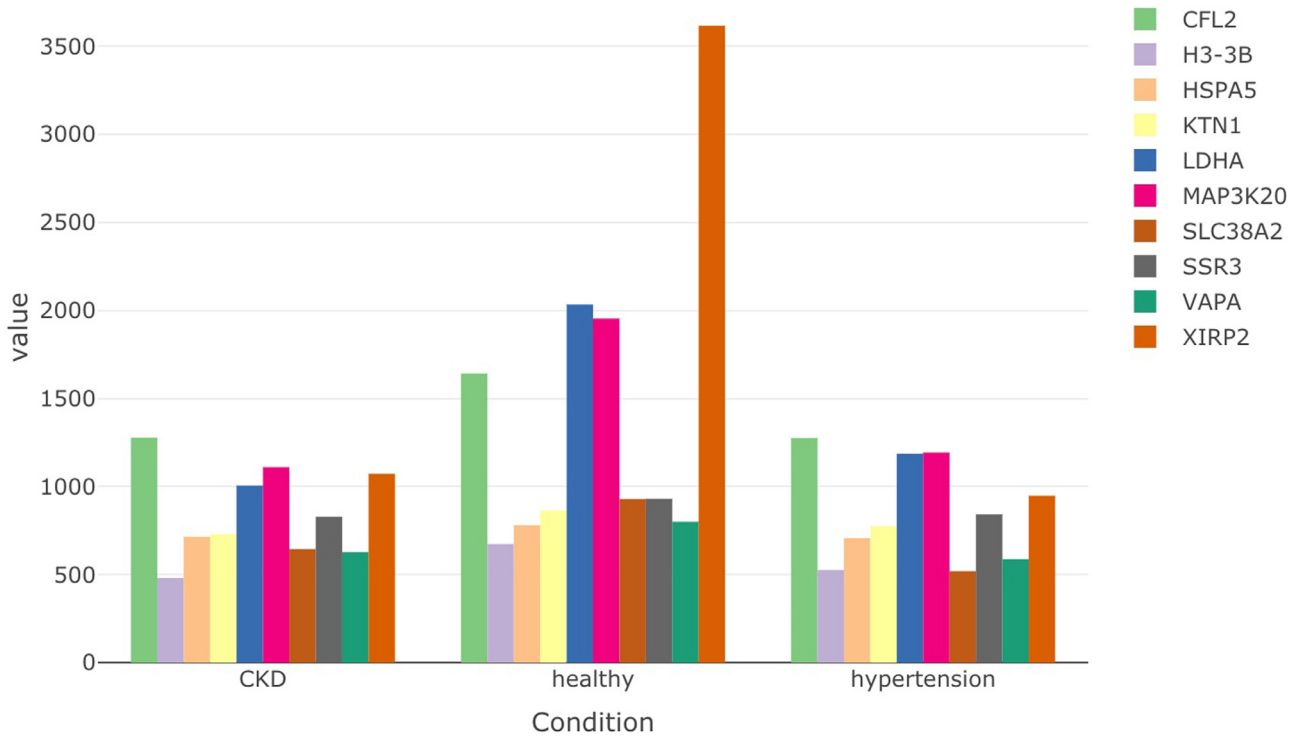

Fig. 12. Individual gene distributions when selecting a specific expression pattern and range (the example is from a "down-up-down" pattern group). CKD – chronic kidney disease patient group, healthy – healthy population group, and hypertension – hypertensive patient group.

```
# Condition subclasses
# "CKD"          "healthy"      "hypertension"

#
# The algorithm can be implemented using a single function call where a meta data file
is also provided to extract information on the subclasses
# pattern_search(data, meta)
#
#               Gene count
#down_down_down          0
#down_down_up         2679
#down_up_down          670
#down_up_up           1076
#up_down_down         5503
#up_down_up           2550
#up_up_down           2856
#up_up_up              361
```

The returned gene list contains groups of genes for the different types of patterns. A pattern of interest can be selected to further explore the genes that changed their expression in a specific manner.

```
# Example pattern selection
#$up_up_down
#   [1] "A4GALT"        "AASDHPPT"      "AATF"
#   [4] "ABCC11"        "ABCC9"         "ABCG2"
#   [7] "ABHD13"        "ABHD2"         "ABI3"
#  [10] "ABI3BP"        "ABITRAM"       "ABO"
```

This analysis can be followed by *pattern_plots* which allows to explore distributions for a selected pattern group. The user must provide a subsetted data frame and low/high parameters to select a specific range. The selection is needed because in some instances the expression values might differ significantly and visualising all data points will prevent exploring any meaningful subsets. The outputs allow to evaluate how genes distribute in a subset for different conditions (Fig. 11) and how individual gene values vary in a selected subgroup (Fig. 12).

Function *cluster_heatmap* uses the information mined from the STRING database [3] to map experimental, referenced, and inferred interactions to see if there are any interactors in the set of significantly changed genes. This heatmap function provides a clustered visualisation of all the genes that have shared interactions (Fig. 13). This information allows to quickly assess how many genes in a specific condition that changed significantly might be part of the same regulatory cluster. Such data can help select specific targets depending on the therapeutic strategy.

```
#Finding interacting proteins and mapping them using a heatmap can be achieved via a
single function call

#cluster_heatmap(data)
```

Function *interactor_map* helps to visualise the information mined from the STRING database [3] and map direct and referenced interactions to see if there are any interactors in the set of significantly changed genes and how they are linked. This visual network is an alternative for a heatmap with additional information on the functional gene features (Fig. 14).

```
#Mapping interactors can be done through a single function call
#interactor_map(data)
```

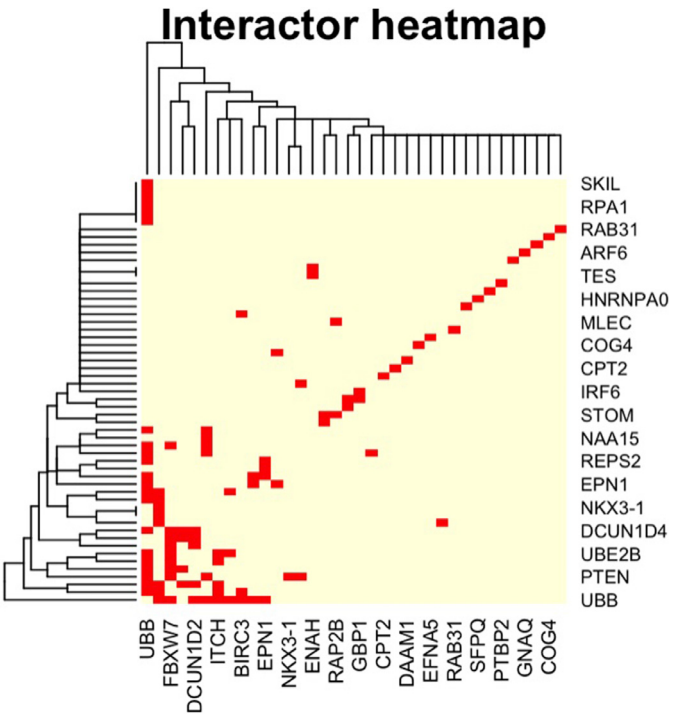

Fig. 13. Cluster heatmap examples where known interactors are connected via the red squares. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

3.4. Epigenomics data integration and analysis

Function *CpG_summary* provides information on the gene CpG island and GC content. The function checks genes against known CpG islands and provides various plots to assess emerging data features. CpG islands were retrieved from the data available with the Genome Reference Consortium (Human Build 38) [17], this information was cross-referenced with the Ensembl database [18] to retrieve overlaps between CpG islands and genes. The function provides a number of analytical plots to assess whether the CpG profile (via GC %) has any influence on the gene expression, interactor number, disease specificity, and disease associations (Figs. 15 and 16). All this information is provided in the context of the assigned protein classes/functional groups. This analysis offers additional insights into the complex interplay between the genome, transcriptome, and epigenome [19]. In addition, the function outputs a data table that contains genomic locations and gene information based on the Ensembl database [18] so that the user can perform additional analyses.

```
#CpG summary requires a single function call to retrieve relevant information. The
output can be assigned to a new R object for further analyses

#cpg_genes<-CpG_summary(data)
```

Function *miRNA_summary_validated* allows to check how many of the differentially expressed genes have known miRNAs (Figs. 17 and 18). The information on validated/known miRNAs is collected from mining multiple databases, namely miRecords, TarBase, miRTarBase, PhenomiR, miR2Disease, Pharmaco-miR. The function also returns a data table with miRNA information that can be used for designing RNA interference experiments.

```
#head(cpg_genes)

#Output example

#   Symbol log2FoldChange       pvalue Association_score
#1   SAR1A      -2.187773 2.656521e-05         0.0000000
#2 C6orf62      -2.674213 1.691567e-07         0.0000000
#3     AXL      -2.786508 1.739595e-04         0.3230769
#4   BICC1      -3.598553 2.738887e-04         0.3000000
#5  CAPZA1      -1.732784 2.321835e-04         0.3789474
#6   TXNIP       1.460629 7.347205e-05         0.3000000
#  Specificity_score  LFCscore                  CpG GC_content
#1             0.000 -2.187773 chr1:1211340:1214153      70.33
#2             0.000 -2.674213                   NA         NA
#3             0.590 -3.686764 chr1:1471765:1497848      58.83
#4             0.751 -4.678119                   NA         NA
#5             0.601 -2.389418                   NA         NA
#6             0.631  1.898818                   NA         NA
#              Class
#1           Receptor
#2                 NA
#3         Pseudogene
#4             Enzyme
#5             Enzyme
#6 Regulatory protein
```

Function *miRNA_summary_predicted* is similar to the earlier function; however, it allows to check how many of the differentially expressed genes have predicted miRNAs. The information is collected from mining multiple databases that use algorithms to infer likely miRNAs. The databases include miRTarBase, PITA, PicTar, miRecords, miRanda, DIANA-microT, miRDB, TarBase, TargetScan, MicroCosm, and ElMMo. The function also returns a data table with miRNA information that can be used in designing RNA interference experiments.

```
#miRNA analysis example
#df<-miRNA_summary_validated(data)
```

Function *miRNA_network* allows to examine if a gene set has shared regulatory miRNAs (Fig. 19).This function could be especially useful as it could help exploring the non-coding layer of the regulatory network. This information can aid in studying how some genes are controlled by several miRNAs and detect additional links between genes that changed expression. Moreover, using miRNA analyses can be applied in designing RNA interference studies to select the most optimal interference sequences. miRNA content information can be access through the function's output.

```
#An example of calling miRNA summary function for the predicted miRNAs based on the
submitted data
#df<-miRNA_summary_predicted(data)
```

4. Discussion

OmicInt package provides a unique combination of functions and tools for researchers to explore gene expression data sets. A special focus of the package is also making machine learning, specifically Gaussian mixture models [4–6], more accessible to the researchers that do not have a background in the ML/AI field. In addition, the lack of tools for the exploration of the complex expressome data highlighted the need for such a set of bioinformatics tools. For example, commercial solutions, such as Clarivate analytics [7], are very expensive and cannot be easily used by individual researchers. Freely available tools, namely GeneMANIA or Cytoscape platforms [9–11,20], do not permit machine learning applications or complex regulome integration. As a result, the *OmicInt* package was developed for advanced and user-friendly *omics* analyses.

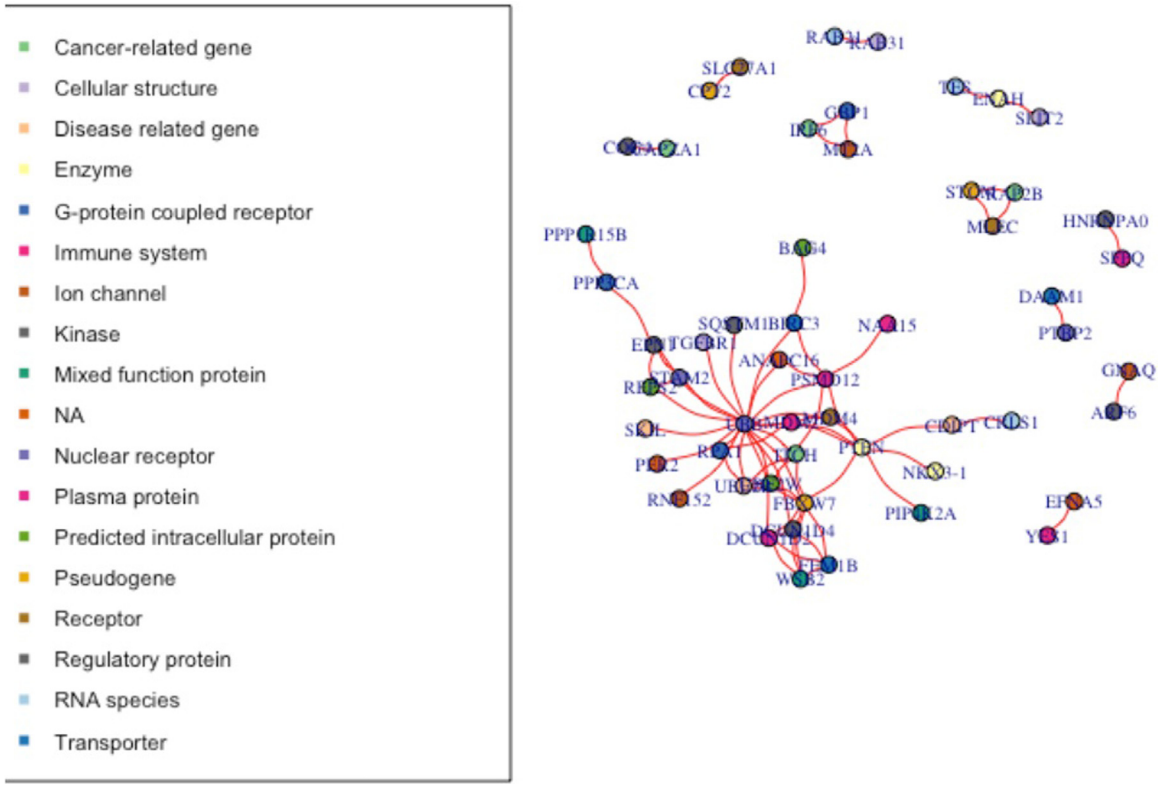

Fig. 14. Interactor map examples.

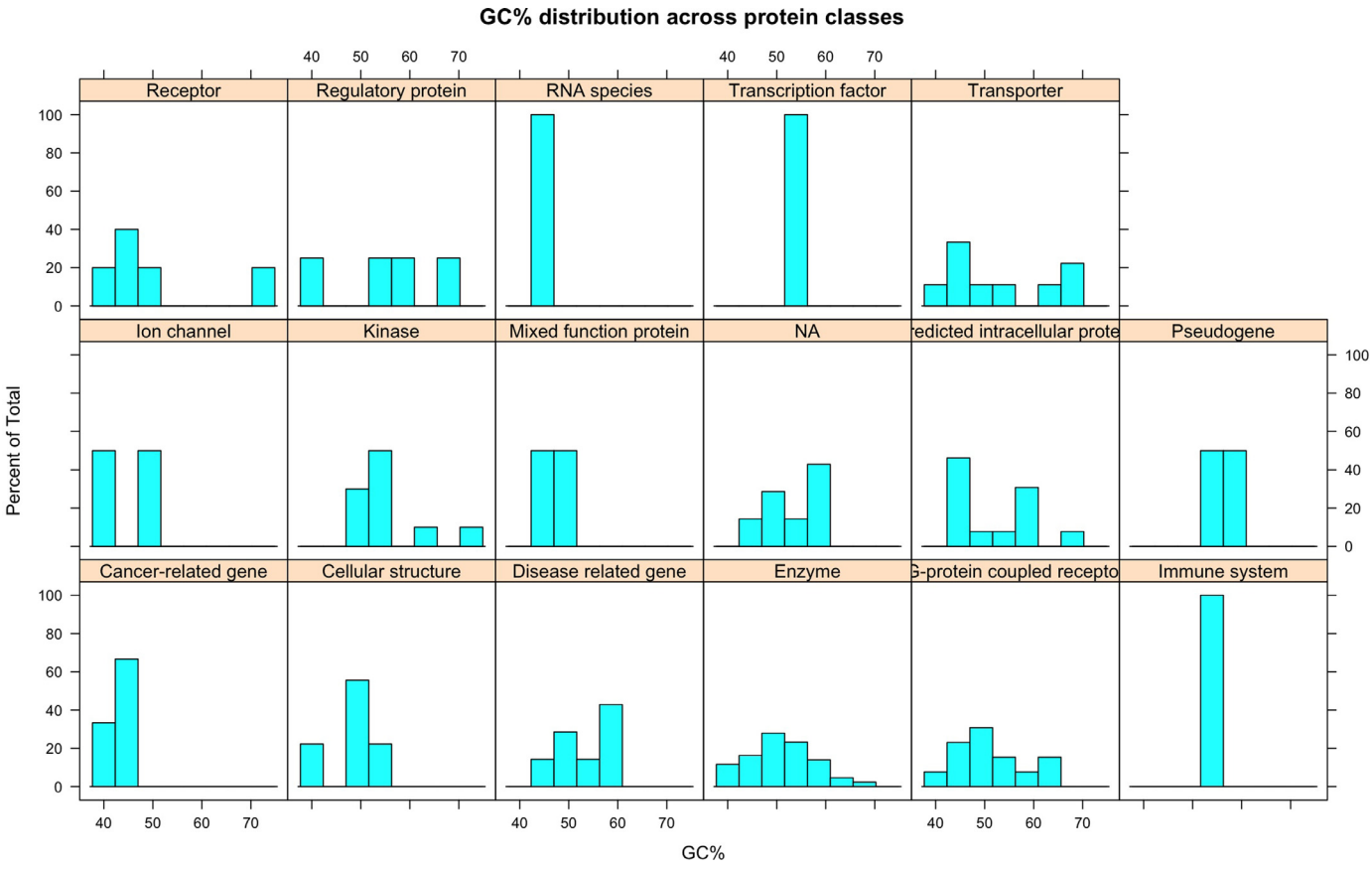

Fig. 15. CpG summary examples where the GC% content distribution is shown for different protein classes.

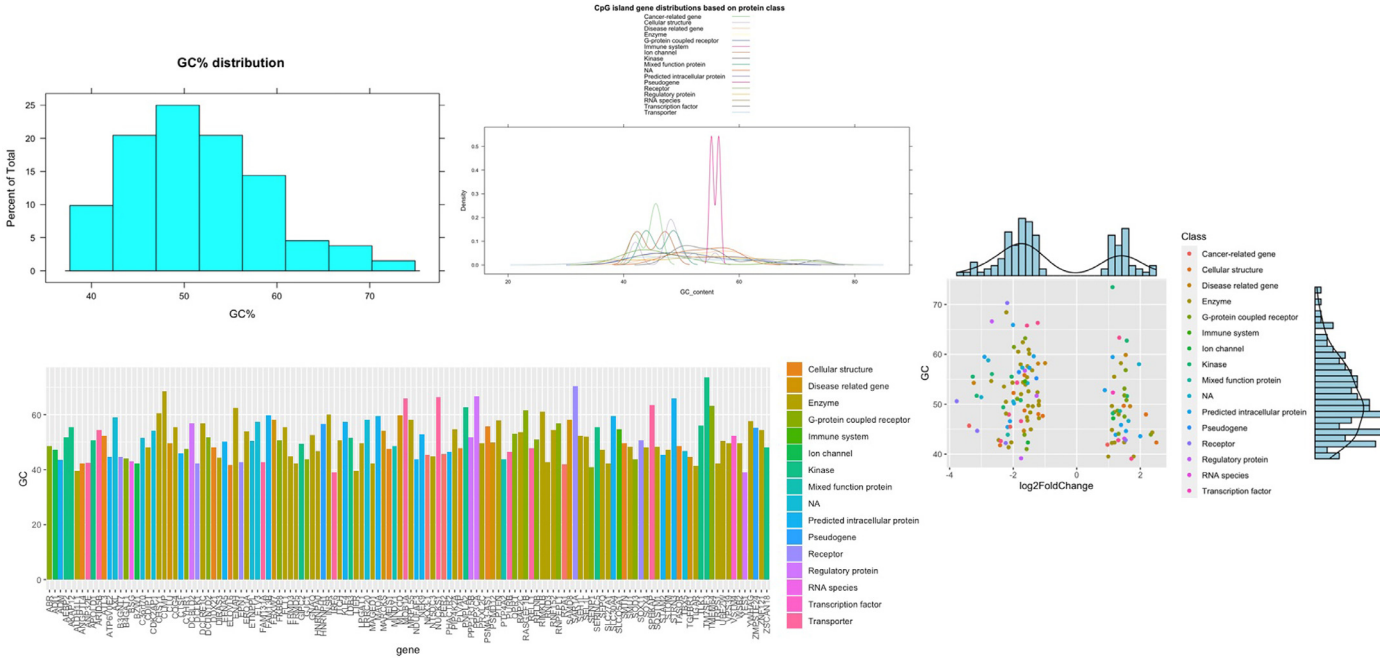

Fig. 16. CpG summary examples where GC% profiles are shown for different genes and their corresponding protein classes. Summary density plots and histograms are also shown for different parameters.

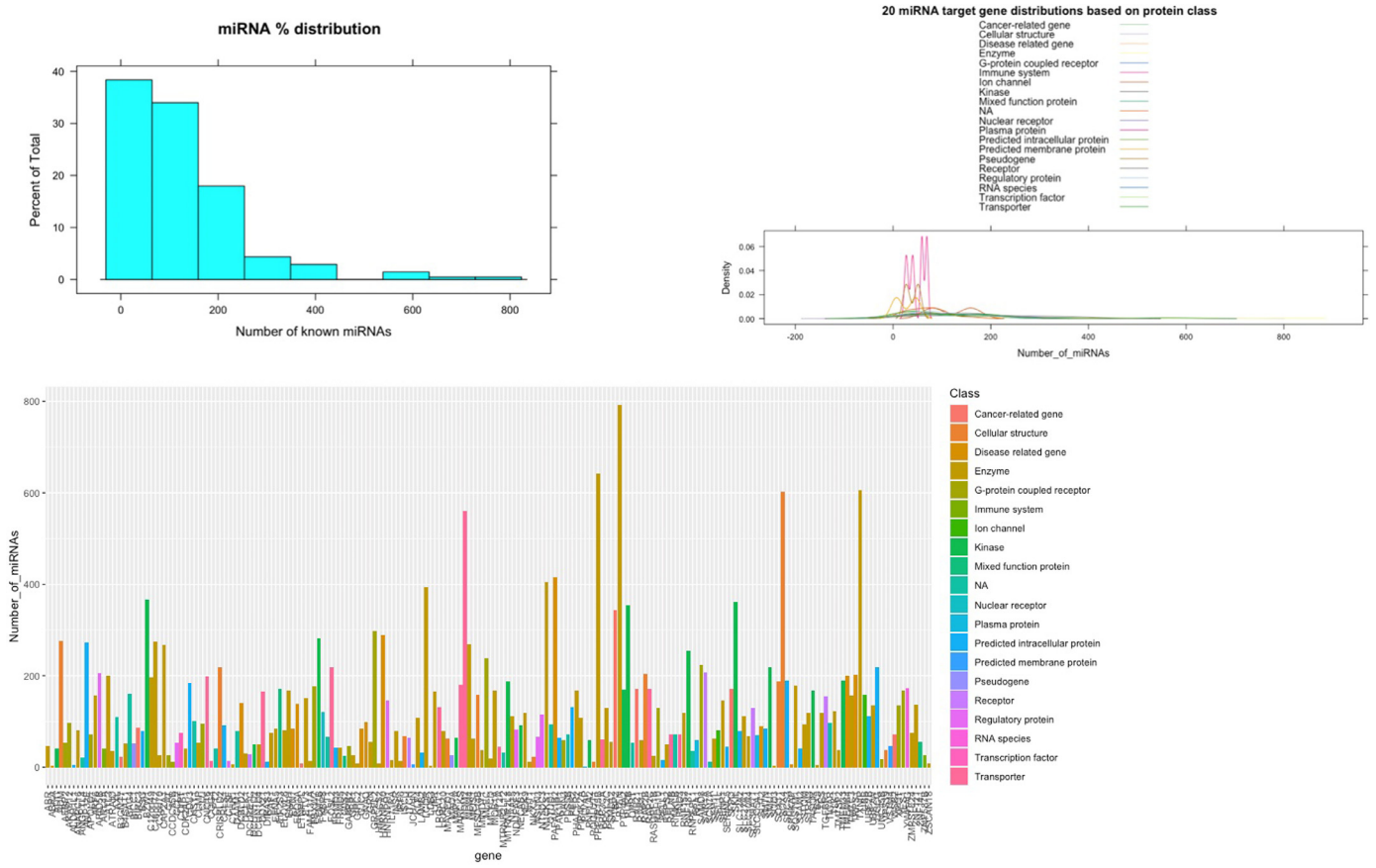

Fig. 17. Validated miRNA summary examples where distribution profiles are shown for different genes and their corresponding protein classes. Summary density plots and histograms are also shown for different parameters.

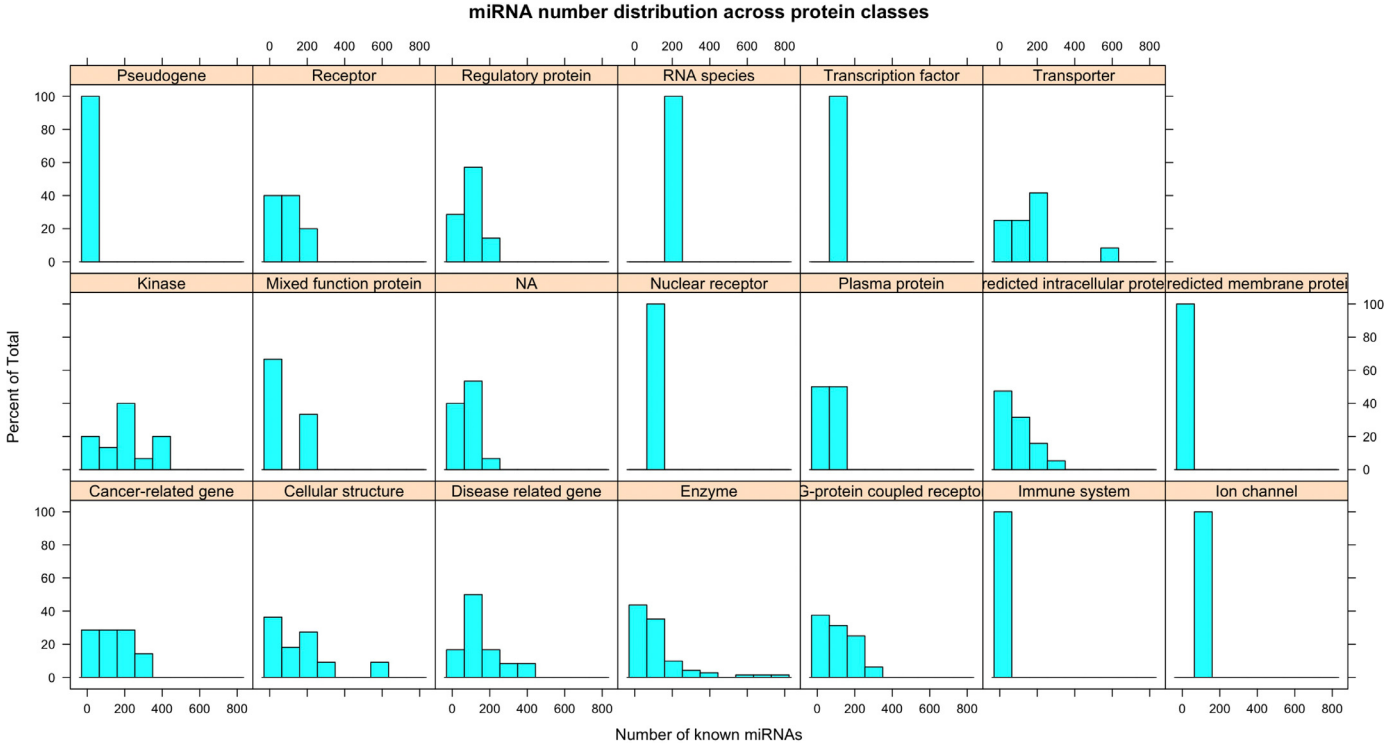

Fig. 18. Validated miRNA summary examples where miRNA content distribution is shown for different protein classes.

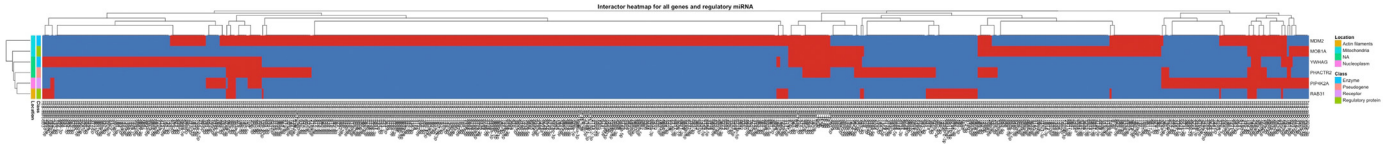

Fig. 19. miRNA network plot example where genes and miRNAs are mapped using a heatmap so that shared links are highlighted in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The developed scoring functions and GMM pipeline enables exhaustive analysis of the expressome and the associated interactome complexity. The automated processing takes care of the machine learning model optimisation making this analysis easily adaptable to individual researcher's needs. The implementation of probabilistic modelling creates opportunities for new insights based on gene expression changes, disease associations, and the size of the network for a specific gene. Extracting this information can establish relevant seed points to recreate complex signalling pathways or use this data to select genes that should be subjected to downstream *in vitro* studies ensuring that a diverse selection is made.

In addition, advanced functions for epigenomics analysis permit the exploration of the epigenetic regulatory layer. This might be very helpful when identifying genes that may depend on epigentic regulation [19]. Specifically, if a CpG island containing gene changed expression during treatment or disease progression, it might suggest that there is an epigenetic component controlling the expression levels. Similarly, exploring a gene's miRNA network could hint at other interacting genes which might not have been picked up by the differential expression analysis or help prepare for RNA interference studies. Moreover, miRNA interactome analysis provides the first in-depth look into what genes are controlled by the same set of miRNAs.

Additional functionalities of the package create an analytical environment to summarise gene functional classes or infer what cellular compartments are typically associated with the gene/protein. Such assessments in the context of expression changes or disease association can highlight emerging patterns in specific cellular states under the investigation. A specially designed function to extract gene pattern profiles can aid in a further refinement of causal gene networks when considering a specific phenotype or a condition.

Thus, *OmicInt* offers a comprehensive, evolving, and adaptable platform for gene expression analysis in the context of the transcriptome, proteome, and epigenome. The analyses are made freely available to all researchers where further contributions and algorithmic development are also made possible.

Declaration of Competing Interest

The author declares no conflict of interest regarding the publication of the manuscript.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.ailsci.2021.100025.

References

[1] UniProt [Internet]. Available from: https://www.uniprot.org/ 2021.
[2] Szklarczyk D, Gable AL, Lyon D, Junge A, Wyder S, Huerta-Cepas J, et al. STRING v11: protein-protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. Nucleic Acids Res. 2019 Jan 8;47(D1):D607–13.
[3] STRING: functional protein association networks [Internet]. Available from: https://string-db.org/ 2021.
[4] Reynolds D. Gaussian mixture models. Encycl. Biom. 2009:659–63.
[5] Kanapeckaitė A, Burokienė N. Insights into therapeutic targets and biomarkers using integrated multi-'omics' approaches for dilated and ischemic cardiomyopathies. Integr. Biol. (Camb.) [Internet]. 2021 May 1;13(5):121–37. Available from: https://pubmed.ncbi.nlm.nih.gov/33969404/.
[6] Liu Z, Song Y, Xie C, Tang Z. A new clustering method of gene expression data based on multivariate Gaussian mixture models. Signal Image Video Process 2015 Feb 8;10(2):359–68. 2015 102 [Internet]Available from: https://link.springer.com/article/10.1007/s11760-015-0749-5.
[7] Clarivate - data, insights and analytics for the innovation lifecycle - Clarivate [Internet]. Available from: https://clarivate.com/ 2021.
[8] Warde-Farley D, Donaldson SL, Comes O, Zuberi K, Badrawi R, Chao P, et al. The GeneMANIA prediction server: biological network integration for gene prioritization and predicting gene function. Nucleic Acids Res. [Internet] 2010 Jul 1;38(suppl_2):W214–20.
[9] GeneMANIA [Internet]. Available from: https://genemania.org/ 2021.
[10] Cytoscape: an open source platform for complex network analysis and visualization [Internet]. Available from: https://cytoscape.org/ 2021.
[11] Otasek D, Morris JH, Bouças J, Pico AR, Demchak B. Cytoscape automation: empowering workflow-based network analysis. Genome Biol. 2019 Sep 2;20(1).
[12] Scrucca L, Fop M, Murphy TB, Raftery AE. mclust 5: clustering, classification and density estimation using gaussian finite mixture models. R. J. [Internet] 2016;8(1):289 Available from: /pmc/articles/PMC5096736/.
[13] DisGeNET - a database of gene-disease associations [Internet]. Available from: https://www.disgenet.org/ 2021.
[14] AusteKan/OmicInt: OmicInt Package [Internet]. Available from: https://github.com/AusteKan/OmicInt 2021.
[15] CRAN - Package OmicInt [Internet]. Available from: https://cran.r-project.org/web/packages/OmicInt/index.html 2021.
[16] Home - GEO - NCBI [Internet]. Available from: https://www.ncbi.nlm.nih.gov/geo/ 2021.
[17] GRCh38 - hg38 - Genome - Assembly - NCBI [Internet]. Available from: https://www.ncbi.nlm.nih.gov/assembly/GCF_000001405.26/ 2021.
[18] Ensembl genome browser [Internet]. 2021. Available from: https://www.ensembl.org/index.html.
[19] Cazaly E, Saad J, Wang W, Heckman C, Ollikainen M, Tang J. Making sense of the epigenome using data integration approaches [Internet]. Front. Pharmacol. Front. Media S.A. 2019;10:126.
[20] Ali M. GeneMANIA: a real-time multiple association network integration algorithm for predicting gene function. 2008;9:1–15. Available from: papers2://publication/uuid/D18A7677-333C-443F-B060-94EFCD7857C6.