Accurate Prediction of Breast Cancer Survival through Coherent Voting Networks with Gene Expression Profiling - Supplementary Materials -

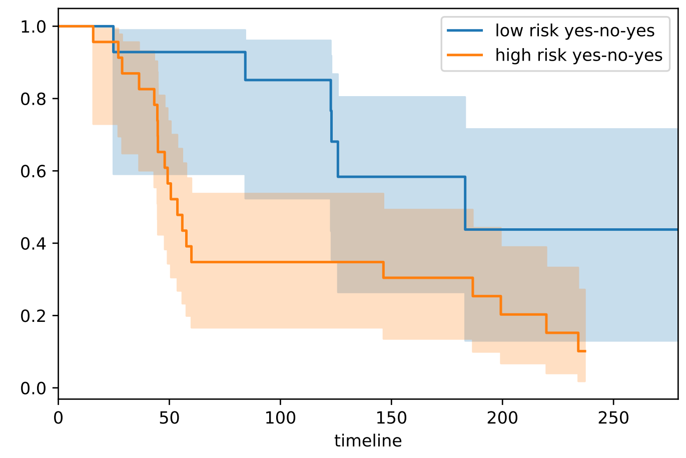
M. Pellegrini
Istituto di Informatica e Telematica del CNR,
Via G. Moruzzi 1, 56100-Pisa (Italy). marco.pellegrini@iit.cnr.it

June 16, 2021

Contents

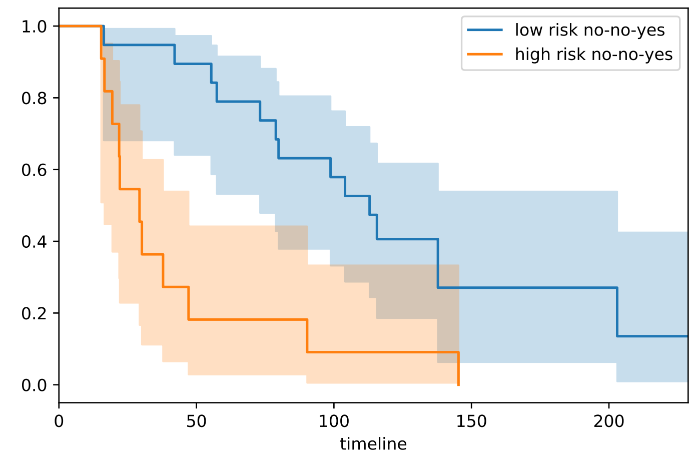
1 Kaplan–Meier plots of therapy-based and secondary stratifications

1.1 Therapy-based stratification



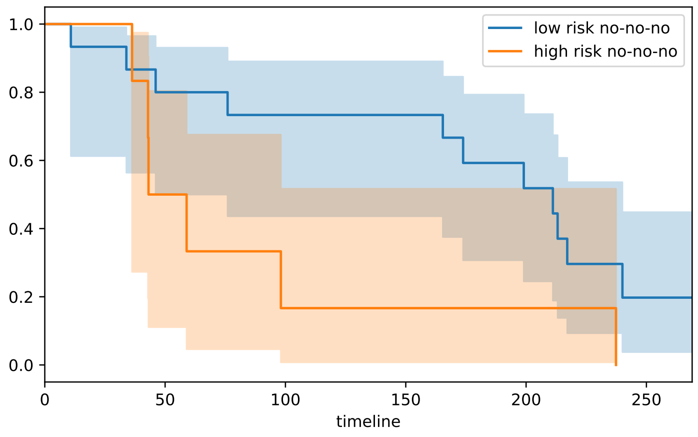
```
yes-no-yes: log rank test p-value 0.022763877908705207
     high risk yes-no-yes   23           13           8            7            4            0            0
      low risk yes-no-yes   14           13          11            4            3            1            0
```

Figure 1: Stratification by therapy class (RAD, CHE, HOR) yes-no-yes.



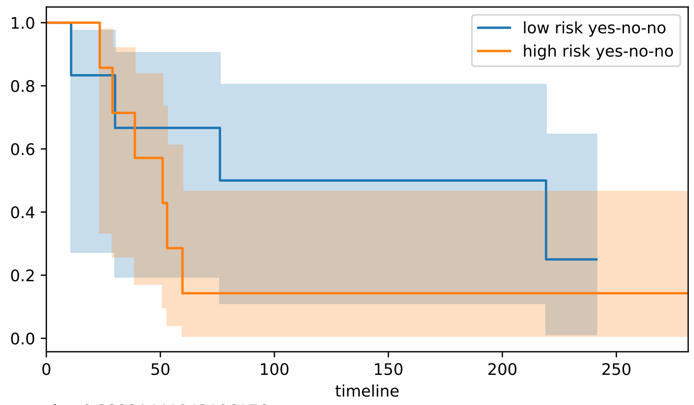
```
no-no-yes: log rank test p-value 0.0006274937044831289
     high risk no-no-yes   11           2           1            0            0            0
      low risk no-no-yes   19          17          11            2            2            0
```

Figure 2: Stratification by therapy class (RAD, CHE, HOR) no-no-yes.



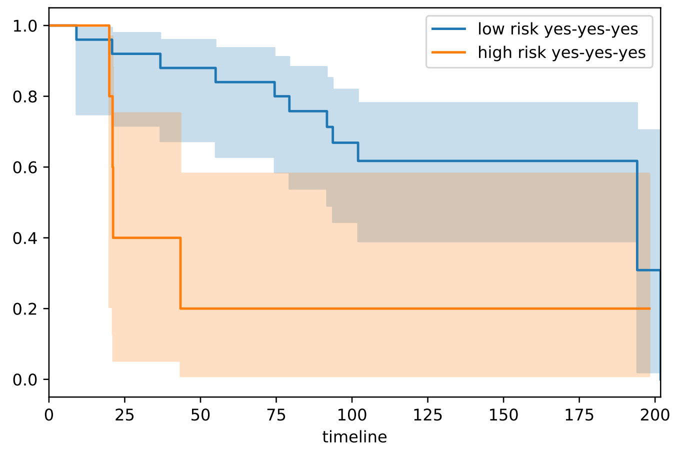
```
no-no-no: log rank test p-value 0.059202578853256194
        high risk no-no-no   6          3          1          1          1          0          0
         low risk no-no-no  15         12         11         11          7          2          0
```

Figure 3: Stratification by therapy class (RAD, CHE, HOR) no-no-no.



```
yes-no-no: log rank test p-value 0.33221441043106176
        high risk yes-no-no  7          4          1          1          1          1          0
         low risk yes-no-no  6          4          3          2          2          0          0
```

Figure 4: Stratification by therapy class (RAD, CHE, HOR) yes-no-no.

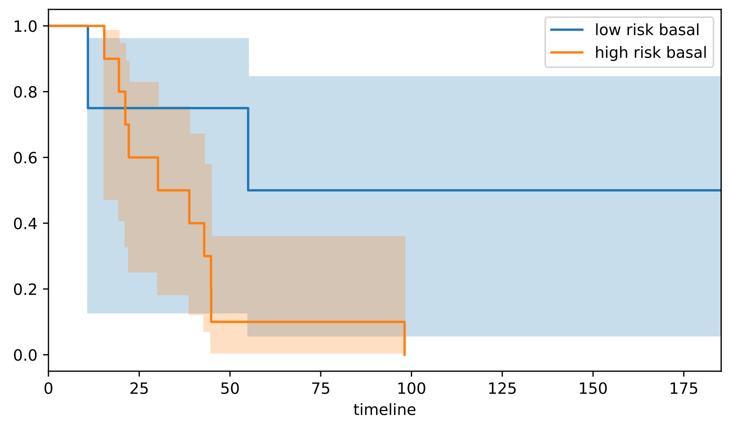


```
yes-yes-yes: log rank test p-value 0.03359476502425298
      high risk yes-yes-yes   5     2     1     1     1     1     1     1     0     0
       low risk yes-yes-yes  25    23    22    20    14     9     6     4     1     0
```

Figure 5: Stratification by therapy class (RAD, CHE, HOR) yes-yes-yes.

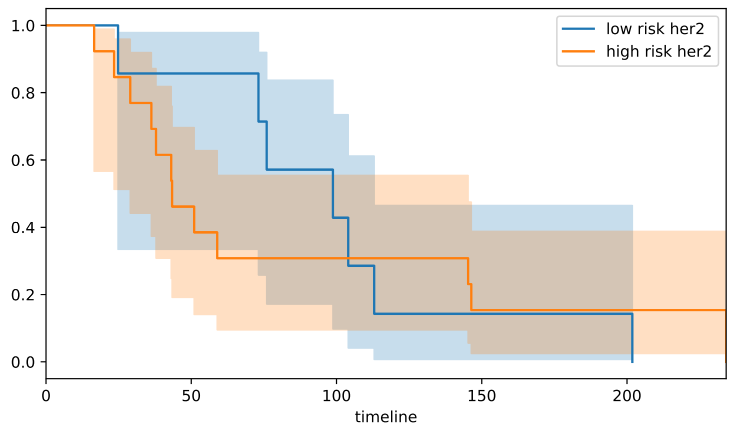
1.2 Secondary stratifications

1.2.1 Intrinsic type



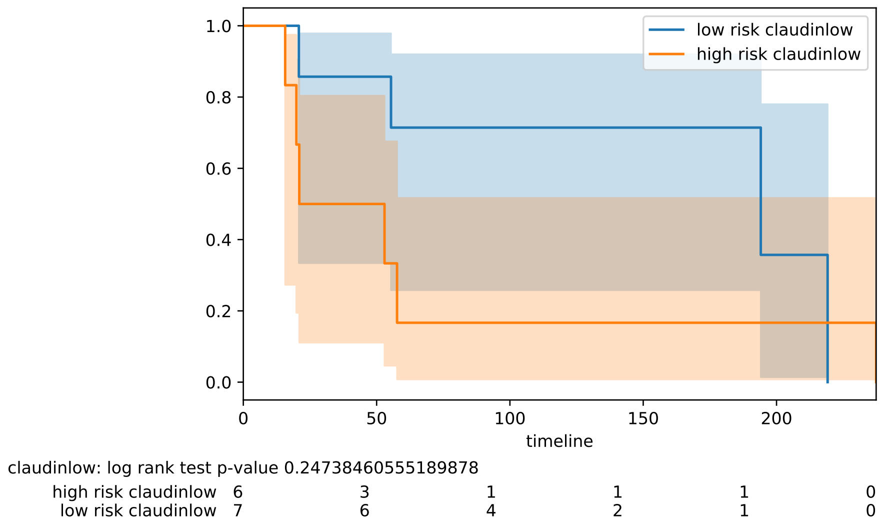
```
basal: log rank test p-value 0.06236264616950541
       high risk basal   10        6        1        1        0        0        0        0        0
        low risk basal    4        3        3        2        2        2        2        1        0
```

Figure 6: Stratification by intrinsic type: basal.



```
her2: log rank test p-value 0.7926616081540957
       high risk her2   13            6            4            2            1            0
        low risk her2    7            6            3            1            1            0
```

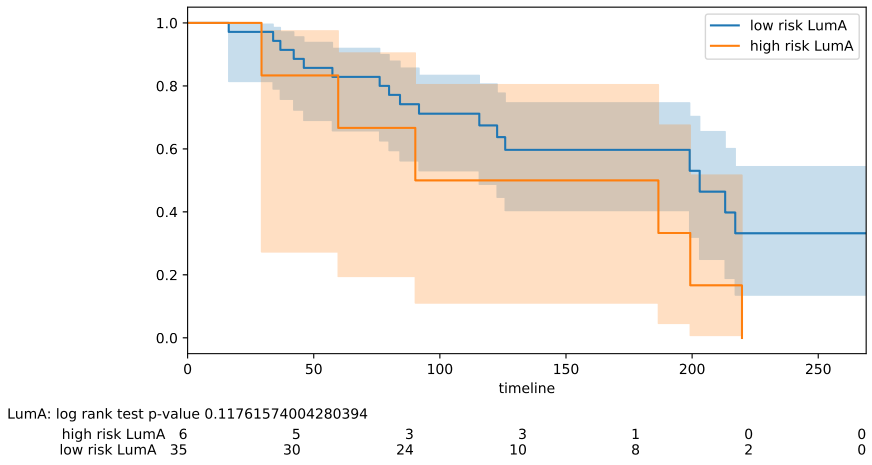
Figure 7: Stratification by intrinsic type: her2.



claudinlow: log rank test p-value 0.24738460555189878

high risk claudinlow	6	3	1	1	1	0
low risk claudinlow	7	6	4	2	1	0

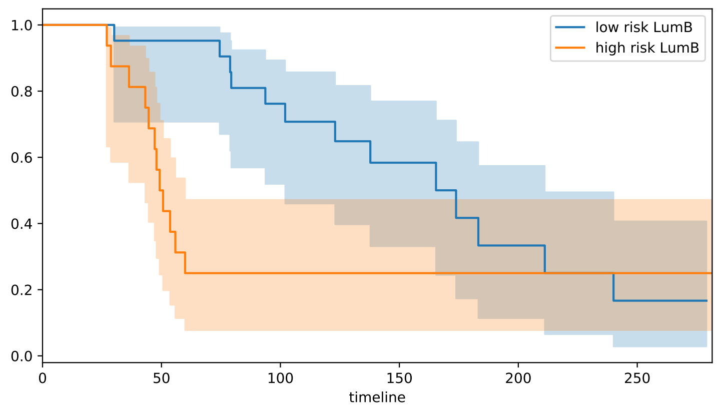
Figure 8: Stratification by intrinsic type: claudin low.



LumA: log rank test p-value 0.11761574004280394

high risk LumA	6	5	3	3	1	0	0
low risk LumA	35	30	24	10	8	2	0

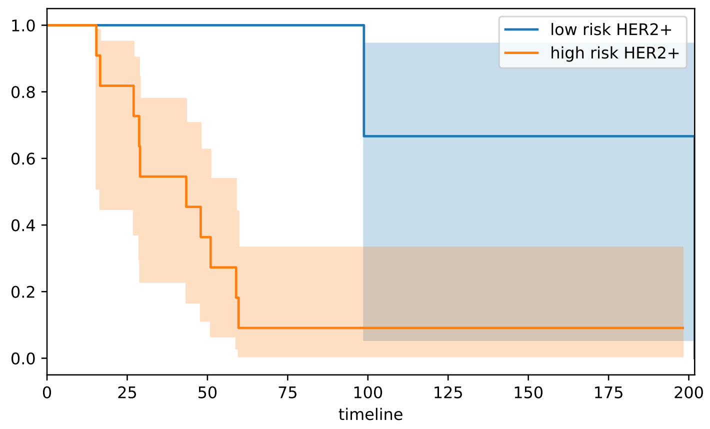
Figure 9: Stratification by intrinsic type: Luminal A.



```
LumB: log rank test p-value 0.057190616538003954
     high risk LumB   16            8            4            4            3            1            0
      low risk LumB   21           20           15            8            4            1            0
```

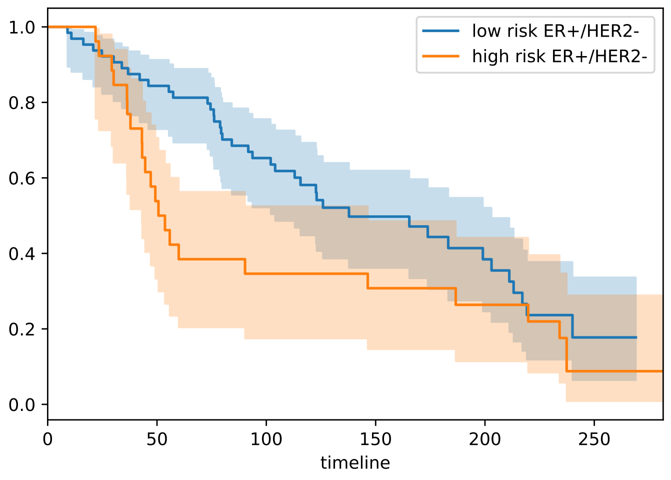
Figure 10: Stratification by intrinsic type: Luminal B.

1.2.2 Hormonal type



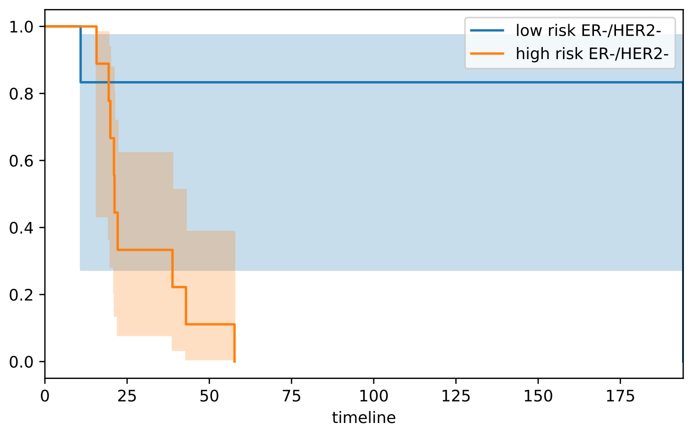
```
HER2+: log rank test p-value 0.013101108556727359
       high risk HER2+   11     9     4     1     2     1     1     1     0     0
        low risk HER2+    4     4     4     4     2     2     1     1     1     0
```

Figure 11: Stratification by hormonal type: Her2+.



```
ER+/HER2-: log rank test p-value 0.07377394696561261
     high risk ER+/HER2-   26        14        9        8        6        1        0
      low risk ER+/HER2-   64        54       40       19       13        2        0
```

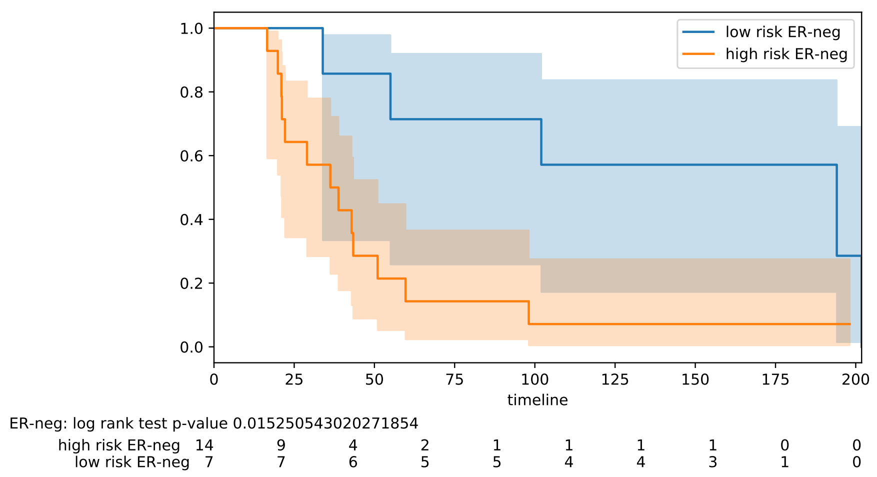
Figure 12: Stratification by hormonal type: ER+/Her2-.



```
ER-/HER2-: log rank test p-value 0.004892913505380565
       high risk ER-/HER2-   9         3         1         0         0         0         0         0         0
        low risk ER-/HER2-   6         5         5         5         4         3         3         2         0
```

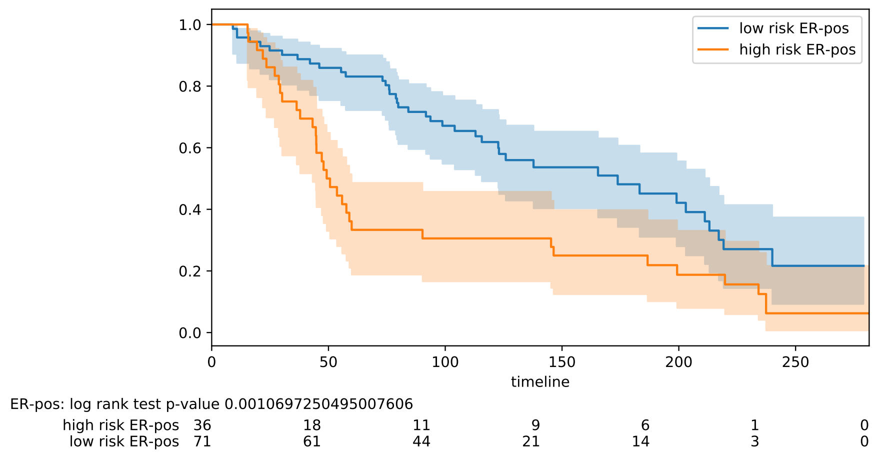
Figure 13: Stratification by hormonal type: ER-/Her2-.

1.2.3 Hormonal ER by IHC



```
ER-neg: log rank test p-value 0.015250543020271854
     high risk ER-neg  14       9        4        2        1        1        1        1        0        0
       low risk ER-neg  7       7        6        5        5        4        4        3        1        0
```

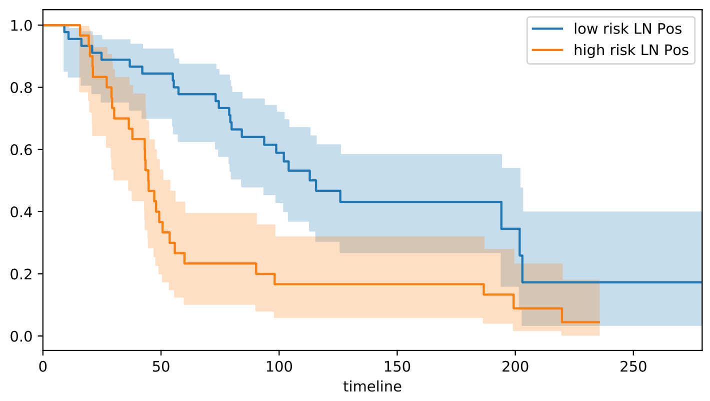
Figure 14: Stratification by IHC ER type: ER-.



```
ER-pos: log rank test p-value 0.0010697250495007606
      high risk ER-pos  36           18           11            9            6            1            0
       low risk ER-pos  71           61           44           21           14            3            0
```

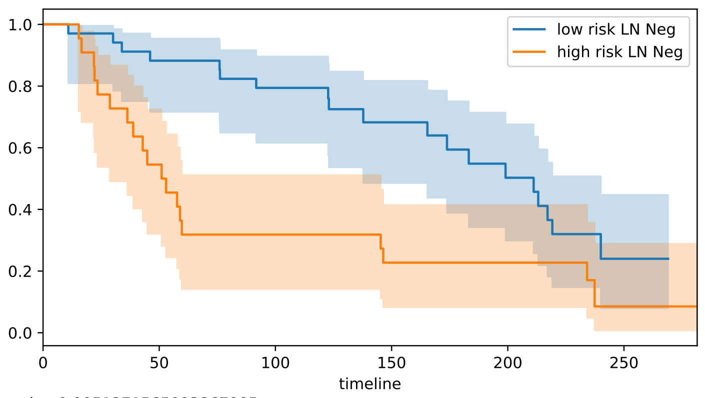
Figure 15: Stratification by IHC ER type: ER+.

1.2.4 Lymph node status



```
LN Pos: log rank test p-value 0.00030867578807149387
        high risk LN Pos   30          11          5          5          2          0          0
         low risk LN Pos   45          38         23          9          4          1          0
```

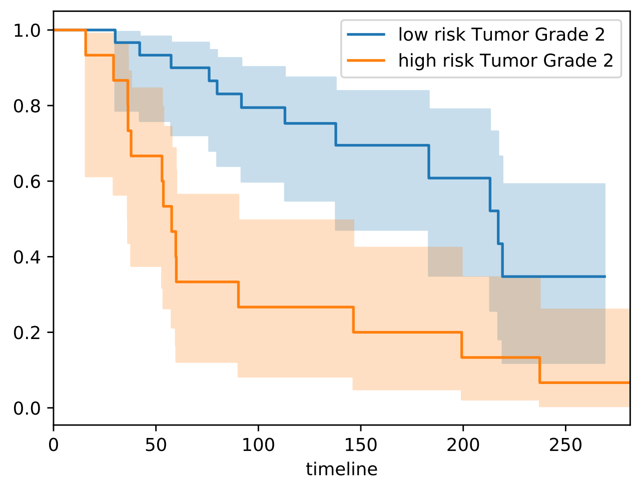
Figure 16: Stratification lymph node status: positive.



```
LN Neg: log rank test p-value 0.0051271565093867905
        high risk LN Neg   22          12          7          5          4          1          0
         low risk LN Neg   34          30         27         16         11          2          0
```

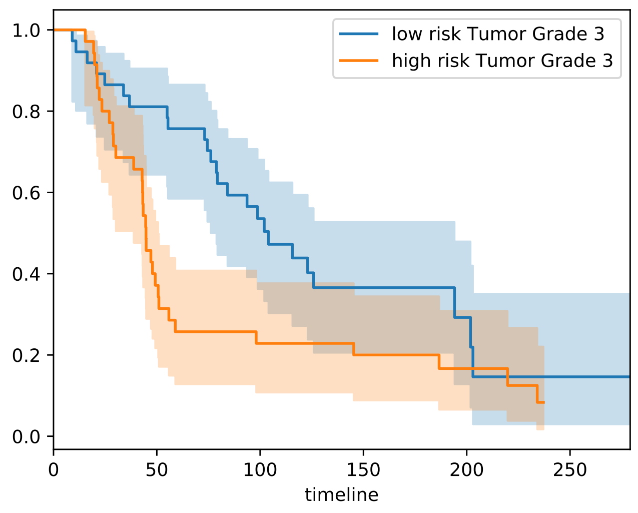
Figure 17: Stratification lymph node status: negative.

1.2.5 Tumor grade status



```
Tumor Grade 2: log rank test p-value 0.0006867222146886424
    high risk Tumor Grade 2  15         10          4          3          2          1          0
     low risk Tumor Grade 2  30         28         22         11          7          1          0
```

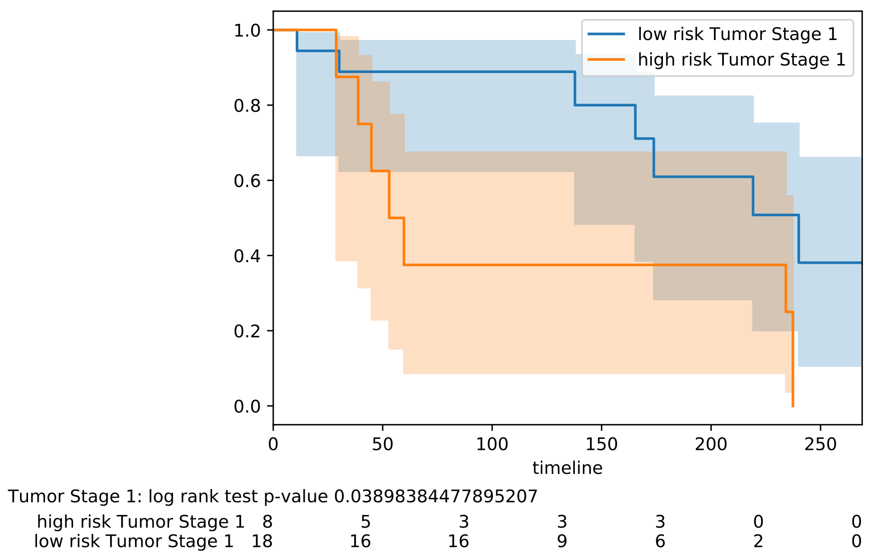
Figure 18: Stratification tumor grade: 2



```
Tumor Grade 3: log rank test p-value 0.022621724759136444
    high risk Tumor Grade 3  35         13          8          7          4          0          0
     low risk Tumor Grade 3  37         30         18          7          4          1          0
```

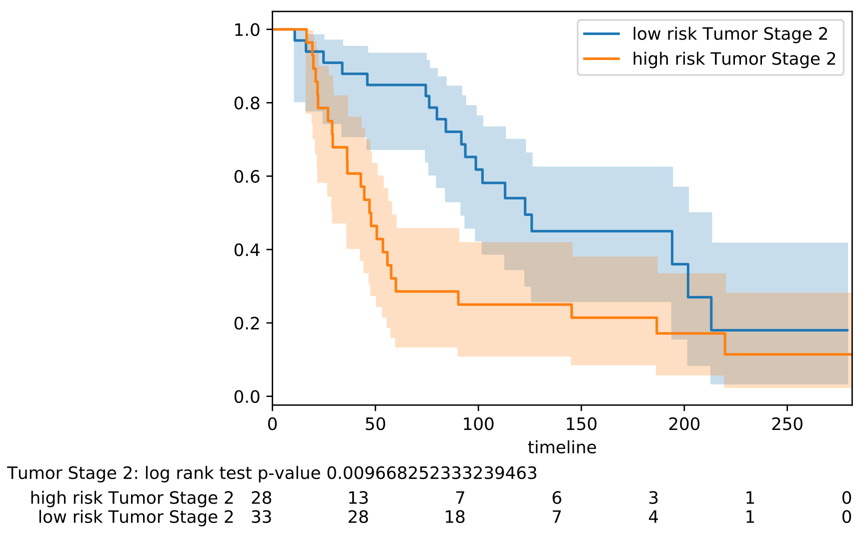
Figure 19: Stratification tumor grade: 3

1.2.6 Tumor stage status



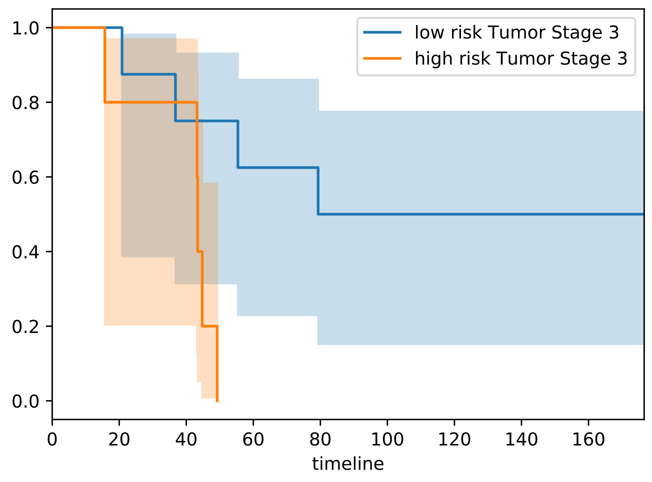
```
Tumor Stage 1: log rank test p-value 0.03898384477895207
   high risk Tumor Stage 1   8          5          3          3          3          0          0
   low risk Tumor Stage 1   18         16         16          9          6          2          0
```

Figure 20: Stratification tumor stage: 1



```
Tumor Stage 2: log rank test p-value 0.009668252333239463
   high risk Tumor Stage 2   28         13          7          6          3          1          0
    low risk Tumor Stage 2   33         28         18          7          4          1          0
```

Figure 21: Stratification tumor stage: 2



Tumor Stage 3: log rank test p-value 0.02094927141077609

high risk Tumor Stage 3	5	4	4	0	0	0	0	0	0	0
low risk Tumor Stage 3	8	8	6	5	4	3	3	2	2	0

Figure 22: Stratification tumor stage: 3

2 Overlap of our fingerprints with previously published multi-gene fingerprints

A comparison of our gene panel and the known 33 multigene panels listed in [1] shows a minimal overlap of our fingerprints with previously described fingerprints for breast cancer. We have 1 overlap over 70 genes for Mammaprint fingerprint (Mamma), and 1 overlap for the 16 genes in the Oncotype DX fingerprint (RS).

Therapy class	gene	Fingerprints
'YES', 'NO', 'YES'	G1	['MBC']
	G2	['p53']
'NO', 'NO', 'YES'	G3	['GGI97', 'Pawitan', 'Robust']
	G4	['Novel2']
	G5	['GGI97', 'Wang']
'NO', 'NO', 'NO'	G6	['IGS', 'MBC']
	G7	['Olaf']
	G8	['MBC']
	G9	['Mamma']
'YES', 'NO', 'NO'	G10	['MBC']
	G11	['HDPP', 'PAM50', 'RS']
'YES', 'YES', 'NO'	-	-
'YES', 'YES', 'YES'	-	-
'NO', 'YES', 'NO'	G12	['Chang']
	G13	['LM', 'MBC']
'NO', 'YES', 'YES'	G14	['Wang']
	G15	['BCI', 'GGI97', 'PAM50', 'Pawitan', 'Robust']

Table 1: Overlap of CVN-based fingerprints and published fingerprints. Fingerprints are indicated with the label used in [1].Common genes are indicated as $G1..G15$.

3 Features of independent cohorts

Many BC data set are available on the NCBI GEO repository. We have selected 4 data sets using the following criteria. The data set should have a sufficient number of patients (at least 30 patients overall). The data set should have survival data (preferably Overalls survival) recorded explicitly for each patient. The data set should have an explicit adjuvant therapy record for each patient, or the same treatment should be declared common to all patients in the data set. We excluded data set for which the treatment record for each patient could be inferred in principle from other records (e.g. hormonal status) but are not declared explicitly. Data from TCGA (The Cancer Genome Atlas) is not used since the treatment status is known only for few patients, and the TCGA (`https://www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga`) records do not follow the same taxonomy of the Metabric records.

4 Measures of performance

Evaluation of prediction performance. For a set of patients whose survival data is known, and a prediction has been computed we have a confusion matrix with entries:

1. TP is the number of patients with actual short survival whose prediction is short survival.
2. TN is the number of patients with actual long survival whose prediction is long survival.
3. FP is the number of patients with actual long survival whose prediction is short survival.
4. FN is the number of patients with actual short survival whose prediction is long survival.
5. NR is the number of patients for which the method does not produce a prediction (no response).

Note that patients for which no prediction is given are not counted in any of the four categories standard categories (TP, TN, FP, FN), thus the number of "no response" (NR) is a further performance parameter to be considered. For the total number of patients examined TOT it hods:

$$TOT = NR + TP + TN + FP + FN$$

Slackness (s).

$$s = \frac{NR}{NR + TP + TN + FP + FN} = \frac{NR}{TOT}$$

The following performance functions are considered:
(Relative) Accuracy (Acc)

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}$$

In situations where the number of patients is fixed and the number of "no answers" is within a small range, we can use also use directly the absolute accuracy (TP+TN) as a figure of merit for ranking solutions.
Positive predictive value (PPV)

$$PPV = \frac{TP}{TP + FP}$$

Negative predictive value (NPV)

$$NPV = \frac{TN}{TN + FN}$$

Sensitivity (Sen.)

$$Sen = \frac{TP}{TP + FN}$$

Specificity (Spe.)

$$Spe = \frac{TN}{TN + FP}$$

Odds ratio (OR):

$$OR = \frac{TP \cdot TN}{\max\{1, FP\} \cdot \max\{1, FN\}}$$

Note that this formulation is always well defined even when FP or FN is zero. For OR we also report the 95% confidence interval and the corresponding Fisher's exact test p-value. Cohen's Kappa (κ).

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

where p_o is the observed relative accuracy of the method, and p_e is the expected relative accuracy of a random predictor that uses the marginal probabilities. Specifically,

$$p_o = \frac{TP + TN}{TP + TN + FP + FN} = ACC$$

$$p_e = \frac{(TP + FN)(TP + FP) + (TN + FN)(TN + FP)}{(TP + TN + FP + FN)^2}$$

Note that the value is undefined when $p_e = 1$, but this case can be attained only trivially when the set of patients has only one class of survival. Cohen's Kappa when positive measures the gain in performance of the proposed classifier against the performance of a random classifier using the marginal probabilities, as measured by its expected accuracy.
The AUC ROC (Area under the curve of the receiver characteristic curve) (`https://it.wikipedia.org/wiki/Receiver\_operating\_characteristic`) measures the average capability of a classifier to discriminate two populations by a scalar value. It can be computed via equivalence to the Wilcoxon Mann Whitney test (`https://en.wikipedia.org/wiki/Mann\_Whitney\_U\_test`).

5 Method in detail: Construction of Coherent Voting Networks

5.1 Definition of the problem, and objectives

We give her details of the methodology for building, training, and using a novel classifier methodology, which we name "Coherent Voting Networks". For concreteness, we describe the technique in the context of the prognostic problem for post-surgery breast cancer patients although its description and scope can be cast in more generic terms.
Given a set of n patients $P = \{p_1, ..p_n\}$, and a set of k genes (mRNA) $G = \{g_1, ...g_k\}$, we have at our disposal gene expression measurements (typically via high throughput transcriptomics) for each gene and in each patient organized as a $n \times k$ matrix $M(P, G)$. For simplicity, we assume a full matrix, however imputation of missing values in the matrix is not needed, as missing entries are just missing edges in the graph representation of the matrix. For each patient we have also the survival function $S : P \to \{\text{low-risk}, \text{high-risk}\}$ where high-risk indicates survival below five years, and low-risk indicates survival above five years. The classification problem we solve is as follows: use the matrix M and the survival function S to train a classifier, so that for a new patient p' for which gene expression levels for a fingerprint $G' \subset G$ of the genes is known, we can infer the value of the survival function $S(p')$ with high accuracy.
Our objective is to obtain simultaneously (a) a fingerprint G' of small size, and (b) a good performance of the classifier as measured in a train-validation-test evaluation pipeline. A train-validation-test, although different from ours, is used as an evaluation protocol of the Sage Bionetworks-DREAM Breast Cancer Prognosis Challenge (BCC) conducted between July 2012 and December 2012.

5.2 Coherent voting networks

The new classification method relies on the hypothesis that the input data (P,G,M,S) admits a combinatorial construction called a "coherent voting network", such construction is in general not unique, thus among the possible constructions we will perform several optimizations and filtering steps for attaining goals (a) and (b).
Definition of a voting network. A "voting network" is composed of (I) a collection $\mathcal{C}$ of subsets of P, $\mathcal{C} = \{C_1, ...C_h\}$ where $C_i \subset P$ for each $i \in [1, h]$ and (II) a pair $F = (F_1, F_2)$ composed of a first level voting function F_1 that maps any restriction of S to a value in $\{\text{low-risk}, \text{high-risk}, \perp\}$, where $\perp$ stands for an undefined label, and a secondary voting function F_2, applied to the output of F_1.
For a patient $p \in P$, let $\mathcal{C}(p)$ be the sub-collection of sets in $\mathcal{C}$ that contain p (i.e. $\mathcal{C}(p) = \{C \in \mathcal{C} | p \in C\}$). Given p, for each set $C \in \mathcal{C}(p)$, the voting function F_1 is evaluated on the restriction of S on $C \setminus \{p\}$ (we do not consider p in the input as a patient cannot vote for herself).

The prediction $Pr(p)$ for p is obtained with the secondary voting function F_2, applied to the values of $F_1(C)$ for all $C \in \mathcal{C}(p)$.
The simplest implementation has both F_1 and F_2 as the majority function (returning $\perp$ in case of tie). However, more complex voting schemes will be discussed below.
The voting network $(\mathcal{C}, F)$ is *coherent* for p if $Pr(p) = S(p)$. We have a *fully coherent voting network* when $(\mathcal{C}, F)$ is coherent for each $p \in P$. We have a *α-coherent voting network* when $(\mathcal{C}, F)$ is coherent for at least a fraction α of the $|P|$ patients.
The reason why this construction is of interest is that, since the vote for a patient p does not depend on $S(p)$, we can take $Pr(p)$ as the prediction for p when $S(p)$ is not know (or not disclosed to the algorithm). A voting network that is coherent, for a high value of α, on its labeled nodes has a good chance of being coherent also for its unlabeled nodes (there will be only one unlabeled node only at any time).

5.3 Construction of a Voting network from the input matrix

The first task is to remove from the matrix M genes that do not have sufficient discriminative power since for these genes random fluctuations and experimental noise (technical or biological) cover the main signal we wish to exploit.
Initial statistical filter on the Gene set. For a gene g_i in G consider the vector V_i(low-risk) of entries of $M(g_i, p)$ when $S(p) =$ low-risk, and V_i(high-risk) of entries of $M(g_i, p)$ when $S(p) =$ high-risk. We apply to V_i(high-risk) and V_i(low-risk) a combination of three elementary statistical tests (t-test, Kolmogorov–Smirnov test, and Mann-Whitney U test), and a fold change test, with given fold change and p-value thresholds. Genes passing the tests are retained in the filtered set $G' \subset G$ and a reduced matrix $M(P, G')$ is obtained. This phase requires the setting of a few parameters, including the type of test performed, the maximum p-value for accepting the test, and the threshold for the accepted on the *fold change.* Here techniques for Differential Gene Expression analysis could be used, such as ANOVA, or tools like DSeq2 [2], edgeR [3]. However, this initial screening phase is in our intention quite light, as subsequent phases should take care of further refining the gene set to attain the final fingerprint.
Gene expression level quantization. Gene expression measurements have a large dynamic range of values, and it is unlikely that two patients have the same expression level of a gene, also due to measuring errors. To define a discrete structure, we quantize the dynamic range of values of each gene into intervals and we replace each entry in M with the corresponding interval. Intervals are induced by cut-points on the real line. We set a minimum and a maximum number of cut points, and a minimum and maximum percentage of patients in each partition induced by the cut-point. We recursively apply cut-point selection methods derived from [4, 5] until the limit of cut points is reached, or some internal stopping criterion is reached. This phase requires the setting of a few parameters, including the type of quantization objective function, the minimum and thee maximum number of cut points selected, and the minimum and maximum percentage of values in either interval generated by a split point, and the number of significant digits to be considered in the gene expression measure-

ments. Also, such quantization functions need to be able to handle patients for which the survival function is undefined.

Patient-gene-interval bipartite graph. After the quantization phase, each entry of the matrix $M(P, G)$ is a pair (gene, interval on the real line). We thus build a bipartite graph in which there is a first class of vertices V_P, labeled with the patient's id. The second class of vertices is $V_{G,I}$ where we have a node labeled with a pair of identifier: the first identifier indicates a gene in G, the second identifier is an interval I. We set an arc from node $p \in V_P$ to node $(g, i) \in V_{G,I}$ if and only if $M[p, g] = i$. We call this bipartite graph $BG(P, G, I)$.

Dense Community detection in bipartite graphs. Similarly to [6], we define a *partial dense cover of radius 2* for the bipartite graph $G(V_1, V_2) = BG(P, G, I)$. A partial dense cover is a collection of dense subgraphs of G with density above a threshold δ and with a number of nodes of type 1 above a minimum threshold q_1 and a number of nodes of type 2 above a minimum threshold q_2. We compute this dense cover using a variant of the *Core & peel* algorithm in [6]. The *Core & peel* algorithm is described in [6] for a graph G, thus we adapt it to handle properly bipartite graph $B = (V_1, V_2, E \subset V_1 \times V_2)$, by changing accordingly the notion of density, and imposing a that a set in the cover has a minimum number of nodes in class V_1 and a minimum number of nodes in class V_2. In our application we will have $V_1 = V_P$ and $V_2 = V_{G,I}$ in the bipartite graph $BG(P, G, I)$ Note that each set of vertices in the cover will have both patients and intervals-gene nodes. Let $\mathcal{D} = \{D_1, .., D_m\}$ be this dense cover. This phase requires the setting of the density threshold (as the percentage of required edges over the maximum possible number of edges on a bipartite complete graph with the same nodes. Typical values range from 0.6 to 0.9. as well as minimum thresholds q_1 and q_2 (usually set to 3).

Determination of a voting network from the partial dense cover. We can obtain from $\mathcal{D}$ a voting network by removing from each $D_i \in \mathcal{D}$ the non-patient nodes (equivalently, projecting the sets in $\mathcal{D}$ on the patient component).

5.4 Sparsification of a partial dense cover

Community sparsification. Our objective in this phase of the algorithm is to obtain from BG a bipartite graph BG' for which we have partial dense cover $\mathcal{D}'$ which is similar to $\mathcal{D}$ for the patient component, but uses a much smaller set G' of genes. Thus we obtain from $\mathcal{D}$ a gene set system GSS by removing from each set in $\mathcal{D}$ the patient-nodes and removing the interval component of the remaining nodes. Next we select a covering threshold c and apply a greedy set multi-cover algorithm [7] to GSS, this algorithm will produce a set of genes G' with the property that for each $g \in GSS$, either $g \subseteq G'$, or $|g \cap G'| \geq c$. Moreover, it has been proved that the number of genes in $|G'|$ is within a logarithmic factor of the smallest subset of G with this property. This phase requires the setting of one main parameter that is the covering number c, typically will be a small number chosen between 3 and 7.

Reduced Patient-gene expression interval bipartite graph. We now compute a reduced bipartite graph $BG(P, G', I)$ using the previously described procedure on the reduced set G', and we re-apply the community detection procedure to obtain the dense cover $\mathcal{D}'$ and the

corresponding voting network. We then test the coherence of the voting network to assess its property of being a α-coherent voting network (for one of the voting functions). If this is verified this voting network can be used for classification purposes.
How to turn a α-coherent voting network into a classifier. Intuitively, we assume that the algorithm A that produces $(\mathcal{C}, F)$ starting from (P,G,M,S) and a fixed vector of parameters is sufficiently stable under small perturbations of the input. We apply the same construction to $(P \cup \{p\},\ G',\ M(P \cup \{p\}, G'),\ S \cup \{p \to \perp\})$. When we control the coherence of the resulting output $(\mathcal{C}', F)$, the level of coherence α' can be computed for all patients in P. We check that $\alpha' \approx \alpha$ to accept $Pr(p)$ as the prediction for the new patient p, based on the coherence of the network on the primitive set of patients.

6 Extending the repertoire of voting schemes

The primary voting function F_1 is chosen among one of the following standard voting functions:

a) "unanimity": the voting function returns label $X \in \{\text{low-risk}, \text{high-risk}\}$ if there is at least one patient with label X and all patient for which the label is not undefined have the same label X.

b) "qualified majority at threshold t:

c) "simple majority": the voting function returns label $X \in \{\text{low-risk}, \text{high-risk}\}$ if there is at least one patient with label X and the number of patients with the same label X is larger than for the number of patients with any other defined label.

d) "more than expected": the voting function returns label $X \in \{\text{low-risk}, \text{high-risk}\}$ if the number of patients with label X is larger than the expected number of patients with label X in a random sample of patients from P, of the same size, with p-value $p \leq 0.05$. This voting function is important when the Training data has unbalanced classes.

e) given the survival function $S : P' \to \{\text{low-risk}, \text{high-risk}\}$ we compute a weighting function on the elements of $\{\text{low-risk}, \text{high-risk}\}$ by counting the number of patients with a given label. Formally the wight of label X is $|\{p \in P' | S(p) = X\}|$

The secondary voting function F_2 is chosen among:

f) "simple majority", when the primary voting function is one of a), b), c), d).

g) "weighted majority": when the primary voting function is e), we sum the weights for each label, and return the label with the highest cumulative weight.

h) "unanimity plus weighted majority": We apply the simple majority on the primary vote done with the "unanimity" function, if the result is defined we return it, otherwise we apply the "weighted majority" and we return this value.

This phase requires the setting of one main parameter that is the voting policy.

7 Method in detail: Training and optimization of Coherent Voting Networks

Train-validation-test evaluation pipeline. The cohort METABTRIC of roughly 2000 patients is split randomly into three sets with proportions (1/2, 1/4, 1/4): training set (roughly 1000 patients), validating set (roughly 500 patients, and testing set (roughly 500 patients). We remove patients for which the survival class cannot be determined. We further stratify each set of patients by the available therapy information. The adjuvant therapy information is whether the patient has received endocrine therapy (YES/NO), radiation therapy (YES/NO), or chemotherapy (YES/NO). This stratification produces eight subclasses of patients. In each subclass of patients, we check the ratio of the size of the two survival classes $\{\text{high-risk}, \text{low-risk}\}$. When the ratio of the largest to the smallest class was greater than 2.5, we use the equalized version of the data.
The train-validation pipeline is applied for each class separately on the corresponding three sets of patients. Let us denote the tree sets of patients for a class as $Train$, Val, and $Test$. We aim at finding a small fingerprint, optimizing the CVN hyperparameters, and evaluating the performance of the chosen classifier. A configuration C is a pair (G', Par) where G' is a set of genes (as obtained by the sparsification procedure) and Par is a vector of parameters that are given as part of the input to the several phases of the algorithm. Each field of Par takes values in a domain of small discrete size. The vector space of all possible parameters is denoted by PAR.
Generation of fingerprints. For a given fixed vector of parameters $Par \in PAR$, we obtain from the input ($Train$, G, $M(Train, G), S(Train)$) a voting network and a panel of genes G' after sparsification. Repeating this operation for all the vectors of parameters in vector space of parameters (PAR) we obtain a pool of candidate gene panels G' (and corresponding vector of parameters and voting networks).
Post-training selection of fingerprints. We aim at reducing the number of candidate gene panels by retaining only a few high-quality candidate gene panels. We discard a candidate panel G' if it is too large (we apply a cutoff value of 20 genes). We discard a candidate panel G' if the quality parameter (α) is below a factor 0.85 of the top value for α among all the generated voting networks. We sort the so filtered fingerprints by their value of α for the associated voting network and we take at most the top ranking 30 fingerprints, after removing eventual duplicate candidate panels.
Validation and Testing. In this phase, we aim at finding a configuration (fingerprint and parameters) the balances the performance on the LOOCV (leave-one-out-cross-validation)

measures for the Train set, on the train-test evaluations of the Validation data, and on the train-test evaluations of the Test data, with minimal usage of Test data (as quantified by the lookahead number).
A straightforward sorting by just one of the quality measure does not give the desired outcome, therefore we use a more complex schema that uses in intermediate steps Pareto frontiers `https://en.wikipedia.org/wiki/Multi-objective\_optimization` and Pareto stratifications of the (multidimensional) performance data vectors, and lower bound filters to discard outliers. For this part of the method by *score* we intend either the kappa statistic or the OR statistic. A configuration is represented by a point in 3D space corresponding to the coordinates (accuracy, score, antislack). Antislack is 1 minus the slack factor.

a) Automatic lower bound computations. Based on Validation data only we compute Pareto strata (iterated the Pareto fronts) up to collecting at least 8 unique points we compute for these points the minimum number of hits (accuracy) and the minimum score, subject to slack below 0.2. These two values (accuracy and score) form the filtering values and are computed on all the Validation data.

b) For each candidate fingerprint separately, we retain the points (accuracy, score, antislack) on Validation data that satisfy the lower bounds, and we compute the principal Pareto frontier of this subset of points.

c) We collect all such Pareto frontiers in a single set, removing duplicate points (i.e. if two configurations correspond to the same point only one is retained).

d) This reduced set is then re-split into Pareto strata but the point associated with a configuration is now based on the LOOCV of Train data of the corresponding configurations and each stratum is sorted by the LOOCV score value. In a variant of this phase, the points may be constructed as coordinate-wise arithmetic mean (or geometric mean) of the LOOCV of Train points, and the corresponding points for the Validation data.

e) In the order of the strata and the internal order of each stratum, we check the stopping criterion. The number of these checks is the *lookahead number* since the stopping criterion involves both Validation and Test data.

f) We stop the lookahead procedure if the Test score value is larger than the Validation score value, or if the Test score value is smaller but within a relative displacement of less than 0.2 from the Validation score value.

8 Manual fingerprint and hyperparameter optimization

For some of the therapy classes, the *Validation and Testing* method outlined above fails to produce a significant result, either because the lower bounding procedure a) is too aggressive, or the stopping criterion in f) does not succeed. For these cases, we resort to a manual selection, by choosing via visual inspection a pool of high-performance configurations on the Validation data. We then *regularize* this set of configurations by taking the Cartesian product of the projections of the configurations on its components. This new set of configurations is then checked on the Test data, and the best performing result on test data is reported. Note that, in this case, the lookahead number is relatively large, in the order of a few tens. However such manually selected fingerprints are of interest since they did perform well on some of the independent sets in LOOCV tests.

9 Avoiding overfitting

When using minimization of cross-validation error to perform hyper-parameter optimization (hypothesis selection), we may incur in a form of ”overfitting”. With a large number of hypothesis and in the presence of noisy data, the minimum of cross-validation error may be attained by fitting the noise in the data, rather than the underlying signal [8]. This phenomenon is discussed by A. Y. Ng [8] who also proposes a principled way of choosing an alternative (non-minimal) hypothesis from the cross-validation error data that has more chances of a better generalization error. Here we do not implement the method by A. Y. Ng, but we observe that in our specific problem a hypothesis that balances well the cross-validation error and the generalization error is found very close to the top of the ranking we build, so that it can be identified with just a few lookahead operations.

10 Data, software, and tools used

The code is written in python 2.7. Kaplan–Meier plots and log-rank tests are produced with the python package *lifelines* (lifelines.readthedocs.io/en/latest/). The ROC AUC and the Odds ratio functions are computed with the python package *scipy.stats* (www.scipy.org/). Autoweka code has been downloaded from github.com/automl/autoweka.
Data pre-processing and post-processing has been done on a Intel Core i7 (2.8 GHz) processor with 6GB RAM and 240 GB HD with Windows 10 Pro (64 bits). Generation and optimization of CVN have been run on Linux server with 8 Intel Xeon E5-4620 v2 @ 2.60GHz (total 64 cores) and 500 Gb RAM with CentOS Linux 7 (Core).
Metabric BC data has been downloaded from cBioPortal (www.cbioportal.org). Data on independent cohorts has been downloaded from the NCBI-GEO portal (www.ncbi.nlm.nih.gov/geo/).
Input data files used in train-validate-testing are available at:
https://github.com/MarcoPellegriniCNR/Coherent-Voting-Network-for-BC-prognosis.

References

[1] Shujun Huang, Leigh Murphy, and Wayne Xu. Genes and functions from breast cancer signatures. *BMC cancer*, 18(1):473, 2018.

[2] Michael I Love, Wolfgang Huber, and Simon Anders. Moderated estimation of fold change and dispersion for rna-seq data with deseq2. *Genome biology*, 15(12):550, 2014.

[3] Mark D Robinson, Davis J McCarthy, and Gordon K Smyth. edger: a bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, 26(1):139–140, 2010.

[4] Usama Fayyad and Keki Irani. Multi-interval discretization of continuous-valued attributes for classification learning. *Proceedings IJCAI*, page 1022–1029, 1993.

[5] Igor Kononenko. On biases in estimating multi-valued attributes. In *Ijcai*, volume 95, pages 1034–1040, 1995.

[6] Marco Pellegrini, Miriam Baglioni, and Filippo Geraci. Protein complex prediction for large protein protein interaction networks with the core&peel method. *BMC bioinformatics*, 17(12):372, 2016.

[7] Vijay V Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013.

[8] Andrew Y. Ng. Preventing ”overfitting” of cross-validation data. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 245–253, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.