Apellis

A small user guide

February 23, 2021

Dimitra-Danai Varsou, MEng

Contents

1 In a nutshell

Apellis (https://apellis.jaqpot.org), is a web-application developed using R and `shiny` package, based on the related Matlab code that can be found in GitHub (https://github.com/DemetraDanae/optimized-read-across). You can find and pull the **Apellis** docker image here: https://hub.docker.com/r/demetradanae/apellis.

Through **Apellis** users can train a predictive model, download it and use it in order to acquire predictions of a toxicity endpoint (numerical/categorical) of interest for a set of engineered nanomaterials (ENMs). **Apellis** incorporates the concepts of grouping, read-across[1] and optimization through Genetic Algorithms (GA). The application consists of two main tabs for model development using one or two similarity criteria respectively. Each of the model development tabs consists of sub-tabs for training and use of the model:

Read-across training In this part, the users can train a toxicity-predictive model through a GA workflow described by Varsou *et al.*(2019).[2] An external validation approach is used to test the proposed read-across methodology, by dividing the full dataset into training and test subsets. This data partitioning can be achieved either by applying a random partition or a partition method (e.g. Kennard-Stone).[3] The training set is used in the GA workflow and determines the optimal set of descriptors and threshold(s) values. Using read-across technique a table that contains all test ENMs with a successful prediction for the toxicity index is presented, a diagram of test ENMs with their neighbors, as well as diagrams and tables containing information for the optimized parameters and model's accuracy.

- Required specifications: This tab contains a series of operational parameters concerning the evolutionary process and the objective function, that should be tuned by the users.
- Probabilities: This tab contains all the necessary probability values that should be tuned for the evolutionary process.

Obtain predictions In this part, the users can either use the trained model from the previous part right after its development, or they can upload and use a model that was trained and saved anytime, for the toxicity prediction of a set of ENMs with unknown toxicity index. After prediction process a table containing the predicted toxicity value for all the unknown ENMs is presented along with the ENMs neighbor diagram.

For more information please refer to the relevant publication: Varsou, Dimitra-Danai, and Haralambos Sarimveis. "Apellis: An online tool for read-across model development." *Computational Toxicology* 17 (2021): 100146.

1.1 About Apelles

Apelles (pronounced *Apellís*) was a renowned painter of ancient Greece. Apelles was probably born at Colophon in Ionia and prospered during the 112th Olympiad (332–329 BC). Apelles allowed the superiority of some of his contemporaries: his portraits were exceptionally realistic, he was praised for his ingenuity and grace and, the simplicity and completeness of his works were remarkable. Apelles' paintings include: *Alexander the Great wielding a thunderbolt*, *Aphrodite Anadyomene*, the *Calumny* etc. Several Italian Renaissance painters were inspired by him and repeated his subjects however, none of his paintings have survived to this day.

2 Background methodology

Apellis is based in a novel read-across methodology for the prediction of toxicity related endpoints of ENMs. This method lies in the interface between the two main read-across approaches, namely the analogue and the grouping methods, and can employ a single or multiple criteria for defining similarities among ENMs. Based on the formulation and the solution of a mathematical optimization problem, the method searches over a space of alternative grouping hypotheses[1], and determines the one providing the most accurate read-across predictions. For the solution of this problem, an innovative GA was developed.

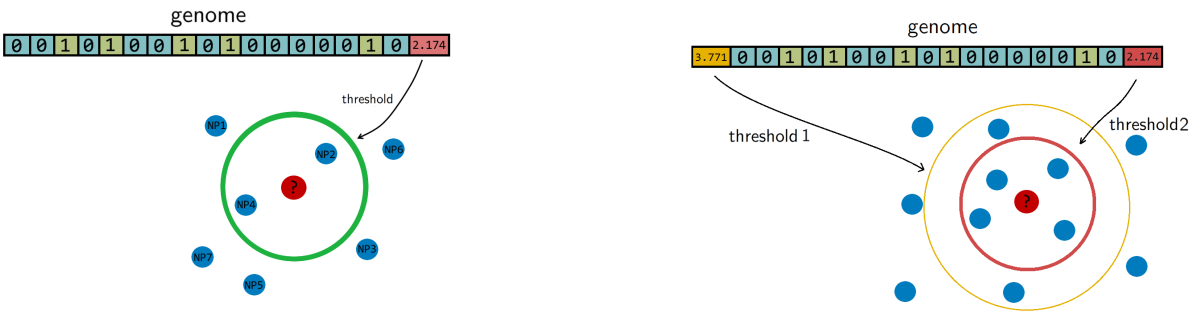

Figure 1: A schematic representation of the proposed read-across approach using the selected variables for determining the neighbors in the multi-dimensional space and for obtaining the final read-across prediction. The optimal threshold value defines a circle around a reference ENM (red particle) and ENMs inside the circle are considered as neighbors (blue particles) whereas the rest ENMs do not belong to the reference ENM neighborhood and are not involved in the read-across prediction. In case of two different thresholds, two circles are defined around each reference ENM and as neighbors are considered only the particles that fall in the inner circle.

In brief, the toxicity endpoint prediction of a target ENM can be performed using the weighted average of the corresponding values of "neighboring" ENMs, i.e. neighboring ENMs of every target ENM are selected by calculating pairwise similarity measures with all available ENMs and by excluding those ENMs for which the similarity measure do not fulfill a predefined threshold. The proposed workflow assumes that a complete dataset is available, i.e. a set of ENMs for which the toxicity endpoint values are known.

If two types of descriptors are available (e.g. physicochemical and biological), similarity measures are calculated between all ENMs separately for the available physicochemical and biological data. For each ENM in the available set, the ENMs for which both similarity measures satisfy the predefined corresponding thresholds, are selected as neighbors. Therefore, in order to characterize two ENMs as similar they need to satisfy both physicochemical and biological similarity criteria. The advantage of this approach is that considers explicitly the multi-perspective characterization of ENMs, by defining multiple thresholds relative to different similarity criteria and ensuring that two ENMs are considered as neighbors only if they satisfy all similarity requirements.

A schematic representation of how the read-across prediction is computed is depicted in Fig.1.

The development of GAs is "bio-inspired" from the principles of species evolution, and is based on the concept that living organisms are examples of successful optimization. The operational parameters of the GAs are summarised next and are directly linked to the biological processes of selection, crossover and mutation of genes:

- Potential solution (*chromosome*): The chromosome contains a sequence of genes with a length equal to the total number of variables.
- Group of potential solutions (*population*): A group of chromosomes (an even number).
- Iterations (*generations*): A number of cycles of selection, crossover and mutation between the potential solutions, leading to an optimal solution.
- Fitness evaluation (*selection*): A process of selection of chromosomes based on their calculated fitness. The reproduction of the fittest chromosomes in the next generation must be assured.
- Combination of two potential solutions (*crossover*): Reproduction operator is employed to exchange genes between two chromosomes, in a random point of crossover.
- Alteration of a potential solution (*mutation*): A process of alteration of the crossovered chromosomes. According to a predefined probability value, the procedure inverts the value of each gene: 0 becomes 1 and *vice versa* (uniform mutation).
- Ensuring desirable evolution (*elitism*): During the creation of a new population with different biological processes, there is a chance of losing the chromosome with the highest score. This method forces the best chromosome to be included in the new population.

- Optimal solution (*genome*): A chromosome containing the combination of genes among the generations that leads to the optimal solution.

The particular GA developed in this work uses the parameters depicted in Table 1 and is explained next. The algorithm is schematically described in Fig. 2.

Table 1: Initialization parameters of the GA

Initial parameter	Details
nChrom	The size of the population, total number of *chromosomes* per generation
generations	The total number of generations
initGeneProb	The probability for a gene to have value 1 initially
crossProb	The probability of crossover
mutProb	The probability for mutation of each gene (uniform)
nonUnf	The mutation probability of the threshold(s) (non-uniform)
$thrGA_{min}$	Lower bound of the threshold(s) value
$thrGA_{max}$	Upper bound of the threshold(s) value
bGA	Freezing parameter
predFactor	Minimum number of samples with produced prediction

Step I An initial population of chromosomes is created. The structure of the chromosomes is shown in Table 2. The chromosome is actually a vector, whose length is equal to the number of descriptors plus the number of similarity criteria used for defining neighbors to a target ENM. The threshold(s) are placed in specific positions in the chromosome representations. This creates hybrid chromosomes containing binary genes for descriptors and real genes for thresholds. A value of 1 means that the corresponding descriptor is selected for defining the distance matrix, while a value of 0 means that the descriptor has not been selected. The probability of a binary gene to be coded as 1 is denoted by *initGeneProb*. The real genes of the chromosomes contain the threshold values corresponding to the similarity criteria and their values are selected randomly from the distance matrices of all samples, considering all variables in each group. In case only one similarity criterion is used, the threshold is placed in the end of the chromosome, whereas if two criteria are used, the two thresholds are placed at the beginning and the end of the chromosome (Table 2).

Table 2: Examples of chromosomes with one and two thresholds

1	0	0	1	0	...	1	1	2.718
2	1.772	1	0	0	...	0	1	1.618

Step II The fitness of each chromosome of the initial population is then calculated as follows:

- The Euclidean distances between all pairs of ENMs are computed using Eq. 1 for a single similarity criterion or Eqs. 2-3 for two similarity criteria.

$$dist_{i,j} = \sqrt{\sum_{\ell=1}^{L} attr_\ell \left(x_{i,\ell} - x_{j,\ell}\right)^2} \quad , i = 1, \dots, N_{tr}, \quad j = 1, \dots, N_{tr}, \quad i \neq j \tag{1}$$

$$distA_{i,j} = \sqrt{\sum_{\ell=1}^{L_A} attrA_\ell \left(xA_{i,\ell} - xA_{j,\ell}\right)^2} \, , i = 1, \dots, N_{tr}, \quad j = 1, \dots, N_{tr}, \quad i \neq j \tag{2}$$

$$distB_{i,j} = \sqrt{\sum_{\ell=1}^{L_B} attrB_\ell \left(xB_{i,\ell} - xB_{j,\ell}\right)^2} \, , i = 1, \dots, N_{tr}, \quad j = 1, \dots, N_{tr}, \quad i \neq j \tag{3}$$

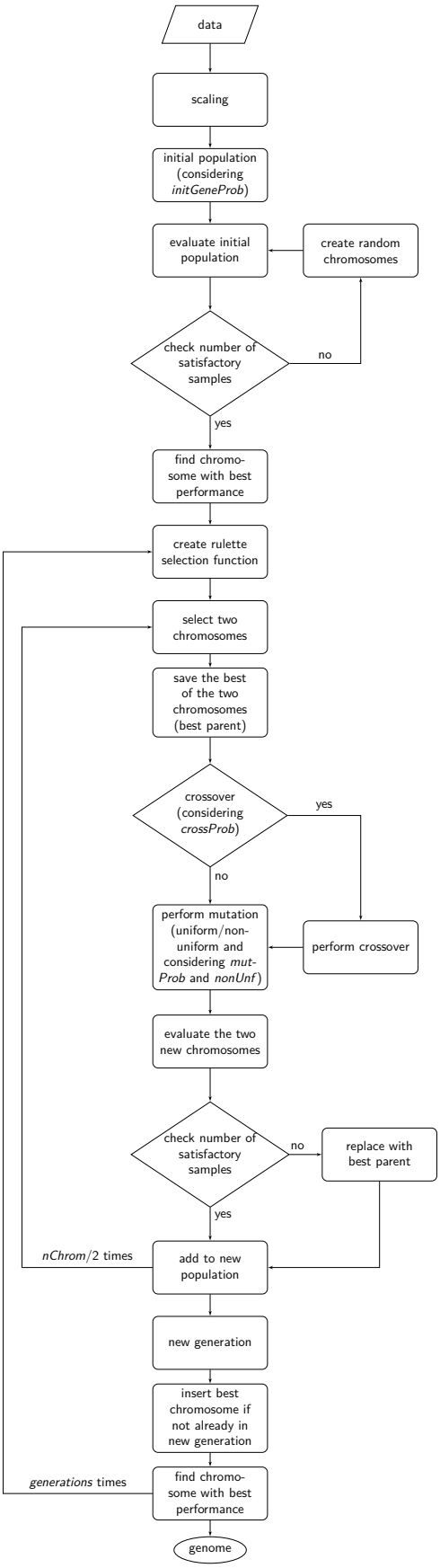

Figure 2: Schematic description of the proposed algorithm.

where, $\mathbf{x}_i = \{x_{i,1}, x_{i,2}, \ldots x_{i,L}\}$, is a vector containing the values of the L descriptors for each one of the N_{tr} available training ENMs and $attr_\ell$, is a binary variable indicating if the descriptor ℓ is selected or not, $\ell = 1, \ldots, L$. Adaptations are made in case that descriptors are grouped into sets A and B containing L_A and L_B descriptors respectively.

- For each ENM, neighbor ENMs are identified as the ones whose distance from the reference ENM is equal or lower than the threshold value (in case of two similarity criteria both distances should be equal or lower than the respective thresholds).
- For ENMs without any neighbor, read-across predictions are not possible. The algorithm checks if Eq. 4 is satisfied, i.e. if ENMs with at least one neighbor are more than $predFactor$ multiplied by the total number N_{tr} of training ENMs. If yes, the algorithm proceeds with next step. If not, the chromosome is rejected, and a new chromosome is generated as described in Step I.

$$\sum_{i=1}^{N_{tr}} pred_i \geq \ predFactor \cdot \ N_{tr} \tag{4}$$

where, $pred_i$ is a binary variable that becomes equal to 1, when a read-across prediction is achieved for the ith ENM, and 0, if no prediction is possible. The total number of ENMs with a successful prediction can be summarized in $N_{pred} = \sum_{i=1}^{N_{tr}} pred_i$.

- The read-across predictions for ENMs with at least one neighbor in the training set are computed using Eq. 5 (Eq. 6 for two similarity criteria).

$$\hat{y}_i = \frac{\sum_{j=1}^{N_{tr}} y_j \cdot \frac{neib_{i,j}}{1+dist_{i,j}}}{\sum_{j=1}^{N_{tr}} \frac{neib_{i,j}}{1+dist_{i,j}}}, \forall i = 1, \ldots, N_{pred} \tag{5}$$

$$\hat{y}_i = \frac{\sum_{j=1}^{N_{tr}} y_j \cdot \frac{neib_{i,j}}{1+distA_{i,j}}}{\sum_{j=1}^{N_{tr}} \frac{neib_{i,j}}{1+distA_{i,j}}}, \forall i = 1, \ldots, N_{pred} \tag{6}$$

where, $\hat{y}_i$, is the predicted endpoint value for the ith training ENM with at least one neighbor, y_j is the endpoint value of the jth training ENM, $neib_{i,j}$ is a binary variable taking the value of 1 if ENMs i and j are neighbors and 0 if they are not and, N_{tr} is the number of training ENMs.

In case of categorical endpoint-prediction the predicted class is estimated by the distance-weighted majority vote of the closest training neighbors, assuming the existence of at least one neighbor for each query ENM (Eq. 7 or Eq. 8 for two similarity criteria).

$$\hat{class}_i = \arg\max_{class} \left(\sum_{j=1}^{N_{tr}} \frac{neib_{i,j}}{1 + dist_{i,j}} \cdot class_j \right), \forall i = 1, \ldots, N_{pred} \tag{7}$$

$$\hat{class}_i = \arg\max_{class} \left(\sum_{j=1}^{N_{tr}} \frac{neib_{i,j}}{1 + distA_{i,j}} \cdot class_j \right), \forall i = 1, \ldots, N_{pred} \tag{8}$$

where, $\hat{class}_i$ is the predicted categorical endpoint value of the ith training ENM with at least one neighbor, $class_j$ the categorical endpoint value of the jth training ENM, $neib_{i,j}$ is a binary variable taking the value of 1 if ENMs i and j are neighbors and 0 if they are not, $dist_{i,j}$ is the Euclidean distance between ENMs i and j and, N_{tr} is the number of training ENMs. All $class$ arguments are of a binary type TRUE/FALSE.

- The mean squared error (MSE) over all ENMs with at least one neighbor is computed using Eq. 9.

$$MSE = \frac{1}{N_{pred}} \sum_{i=1}^{N_{tr}} pred_i \, (y_i - \hat{y}_i)^2 \tag{9}$$

  In case of categorical endpoint-prediction instead of MSE, the Matthews correlation coefficient (MCC) is calculated according to Eq. 10.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{10}$$

  where, TP (true positive) is the frequency of class TRUE ENMs correctly classified as "TRUE", TN (true negative) is the frequency of class FALSE ENMs correctly classified as "FALSE", FP (false positive - Type I error) is the frequency of class FALSE ENMs incorrectly classified as "TRUE" and FN (false negative - Type II error) is the frequency of class TRUE ENMs correctly classified as "FALSE".

- The fitness function value of the chromosome is computed by Eq. 14. Score function has two components the error term (better predictions lead to higher scores) and a regularization term, accompanied by a regularization factor wf_{OF}, that controls the influence of the number of selected variables on the final score. The error depends on the MSE (Eq. 11) or MCC values (Eq. 12), according to the predicted endpoint (numerical or categorical respectively).

  In case that no neighbors are found for all test ENMs, MSE or MCC cannot be calculated and automatically error is equal to Inf. In addition for categorical endpoints, when MCC is equal to -1 (total disagreement between prediction and observation), error is automatically equal to Inf. Infinity in this case is considered equivalent to a large number.

$$error = \begin{cases} \infty & \text{if MSE} = \text{NA} \\ MSE & \text{otherwise} \end{cases} \tag{11}$$

$$error = \begin{cases} \infty & \text{if MCC} = \text{-1 or MCC} = \text{NA} \\ |0.5 - 1/(1 + MCC)| & \text{otherwise} \end{cases} \tag{12}$$

$$OF = error + wf_{OF} \cdot \sum_{\ell=1}^{L} attr_\ell \tag{13}$$

$$score = \begin{cases} 0 & \text{if } error = \infty \\ 1/(OF + 10^{-5}) & \text{if } error \neq \infty \end{cases} \tag{14}$$

- The chromosome with the highest (*best*) calculated fitness is saved for later analysis.

Step III A natural selection process takes place and it is iterated *generations* times. During each iteration, the following procedure is repeated *nChrom*/2 times and in total *nChrom* are selected that form the new generation.

- In order to assure the reproduction of the fittest chromosomes, a "roulette wheel" approach is used. The method selects a pair of chromosomes from the previous population, based on randomly generated numbers that indicate the "slots" corresponding to the different chromosomes. The roulette wheel is constructed so that the size of each slot is proportional to the fitness of the corresponding chromosome. The roulette is "biased", thus chromosomes with a reproductive advantage (better fitness scores), have higher probability to be selected. For each pair of selected chromosomes, the one with the highest score is saved as the *bestParent* for later use.

- The genetic operators of crossover are applied. According to the *crossProb* value, it is decided if the chromosomes are going to exchange strings of genes or not, in a randomly selected point that indicates the position of crossover.
- The genetic operator of mutation is applied. With probability *mutProb*, binary genes that corresponds to a descriptor, invert their value from 0 to 1 and vice versa, while non-uniform mutation is always performed to the threshold values, according to Eq.15.

$$thr_{new}^{GA} = \begin{cases} thr_{old}^{GA} + (thr_{max}^{GA} - thr_{old}^{GA}) \cdot (1 - r^{(1-g/generations)bGA}) \\ \text{if a random digit is 0} \\ \\ thr_{old}^{GA} - (thr_{old}^{GA} - thr_{min}^{GA}) \cdot (1 - r^{(1-g/generations)bGA}) \\ \text{if a random digit is 1} \end{cases} \tag{15}$$

In Eq.15, thr_{old}^{GA} is the old threshold value, thr_{new}^{GA} the threshold value that results from the non-uniform mutation, thr_{max}^{GA} and thr_{min}^{GA} are the upper and the lower bounds of the threshold values, r is a random number between 0 and 1, g is the number of the current generation and, bGA is a parameter which determines the degree of dependency on the generation number.

The non-uniform mutation process, searches the space uniformly in the first place avoiding stagnating, and as the number of iterations approximates the maximum number of generations, convergence is achieved.[4]

- The two new chromosomes are evaluated with the procedure described in Step II and in case a chromosome does not meet constraint 4, it is replaced by its *bestParent*.

In case the best chromosome of the previous generation is not included in the new generation, the algorithm places it in the position of the chromosome with the minimum score, in order to ensure that the chromosome with the best performance will always survive in the evolutionary procedure.

The best chromosome of the last generation is the result of the algorithm. The selected variables and threshold(s)) corresponding to this chromosome will be used subsequently for read-across predictions of unknown ENMs. For evaluating the method, all the training examples are passed through step II described above to produce the read-across predictions.

3 Validation

An external validation scheme [6] is used to test the performance of the produced read-across model in terms of predicting accurately the endpoint on ENMs that have not been used during the training process. To this end, the read-across model which is the final outcome of the training evolutionary workflow, is used to compute endpoint predictions for the ENMs belonging to the test set. These calculations are performed using Eqs. 5 and 7, adapted for test samples. [2]

For numerical endpoints, the following validation metrics are computed: the q_{ext}^2 statistic (Eq. 16) [6], the mean absolute error (*MAE*, Eq. 17) and the *MSE* metric (Eq. 18), which is adapted to the test samples:

$$q_{ext}^2 = 1 - \frac{\sum_{i=1}^{N_{pred}} (y_i - \widehat{y_i})^2}{\sum_{i=1}^{N_{pred}} (y_i - \bar{y}_{tr})^2} \tag{16}$$

$$MAE = \frac{1}{N_{pred}} \sum_{i=1}^{N_{test}} |pred_i (y_i - \hat{y}_i)| \tag{17}$$

$$MSE = \frac{1}{N_{pred}} \sum_{i=1}^{N_{test}} pred_i \, (y_i - \hat{y}_i)^2 \tag{18}$$

where, y_i and $\hat{y}_i$ are the actual and predicted endpoint values over the test set,
$\bar{y}_{tr}$ is the averaged value of the endpoint over the N_{tr} training ENMs and,
N_{test}, is the number of ENMs in the test set ,
N_{pred} is the number of test ENMs with $pred_i \neq 0$.

In case of categorical endpoints, validation results are displayed in a confusion matrix (Table 3), where TP, TN, FP, FN have been defined in Eq. 10. The proportion of actual TRUE-class ENMs that are correctly classified as "TRUE" is measured by sensitivity (Sn, Eq. 19) and the proportion of actual FALSE-class ENMs that are correctly classified as "FALSE" is measured by specificity (Sp, Eq. 20). The overall success rate is measured by accuracy (Ac, Eq . 21).

Table 3: Confusion matrix

		Predicted class	
		TRUE	FALSE
Actual class	TRUE	TP	FN
	FALSE	FP	TN

$$Sn = \frac{TP}{TP + FN} \tag{19}$$

$$Sp = \frac{TN}{TN + FP} \tag{20}$$

$$Ac = \frac{TP + TN}{TP + TN + FP + FN} \tag{21}$$

The MCC metric is also calculated for the test set, according to the Eq. 10, which is adapted for the test samples.

Instructions

Apellis offers two alternatives for model training depending on the available data. In case that only one type of descriptors is available **Numerical single criterion** tab (**Class single criterion** tab for categorical endpoints) must be used, whereas in case of two different types of available descriptors **Numerical multiple criteria** tab (**Class multiple criteria** tab for categorical endpoints) should be used however, single criterion tabs can also be used and all descriptors will be treated as one category. All tabs have similar use, as described in the following paragraphs. **Apellis**' homepage (Fig. 3) includes all the necessary information about its functionality (user guide, video tutorial, information about maintenance and license).

🚲 It is advised to refresh the app when new datasets are uploaded or different tabs are used.

4 Numerical single criterion

This part (Fig. 4) can be used for training and using a read-across model for the prediction of toxicity endpoints and other properties of ENMs, when only one type of descriptors is available (or users aspire to treat the descriptors as one category).

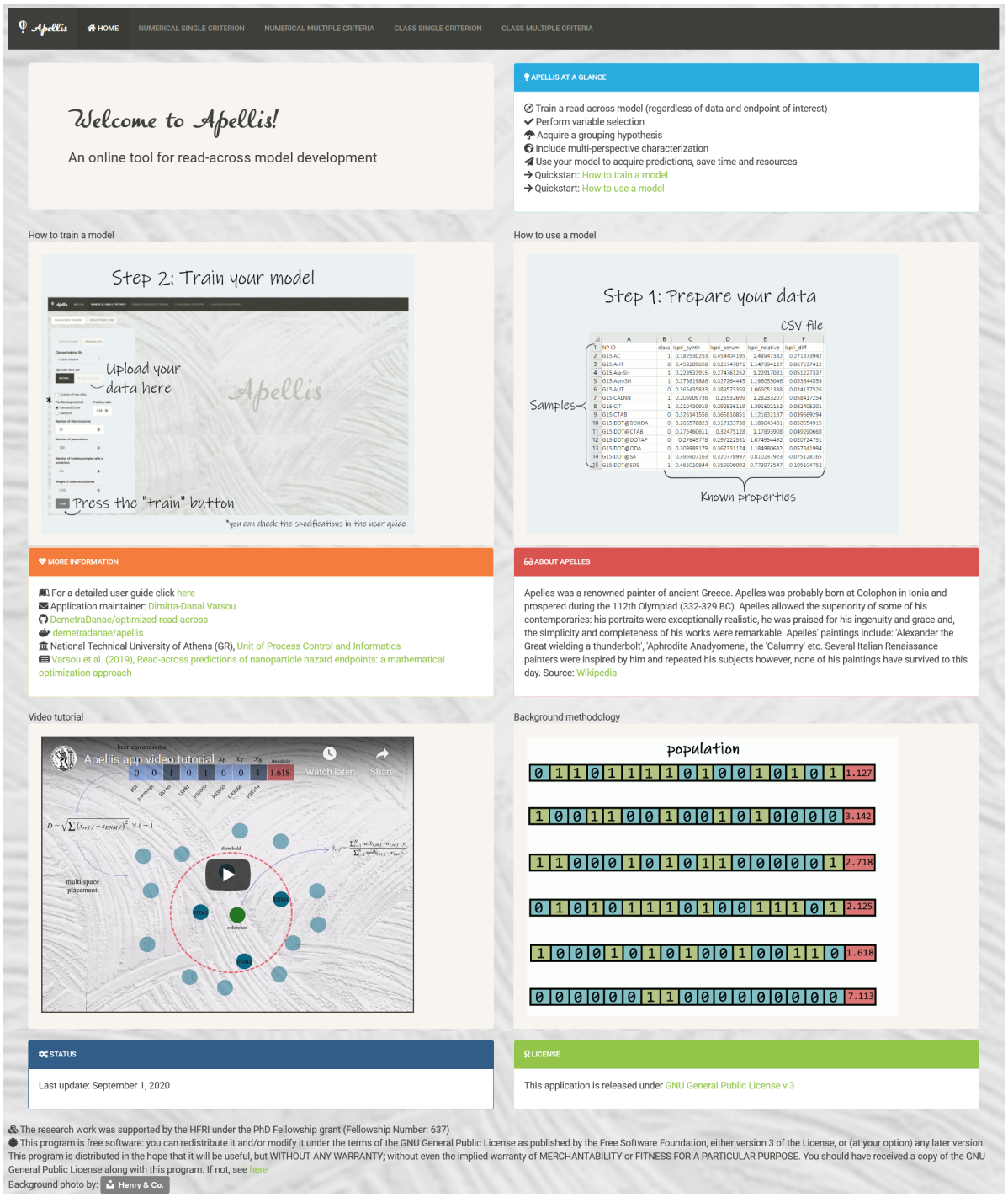

Figure 3: Apellis' landing page. From top ribbon users can chose numerical/class single criterion or multiple criteria model development according to the available data. The landing page includes a summary of the application functionalities and two quick-start guides in *Apellis at a glance* box and useful information about its use in *Help* box. The *Status* box includes the last update date and the *License* box the link towards the license file. In addition, two gifs explain in a few steps how to train and use a read-across model through the app. Finally, a video tutorial for Apellis' use and a gif explaining the background methodology are presented in the landing page.

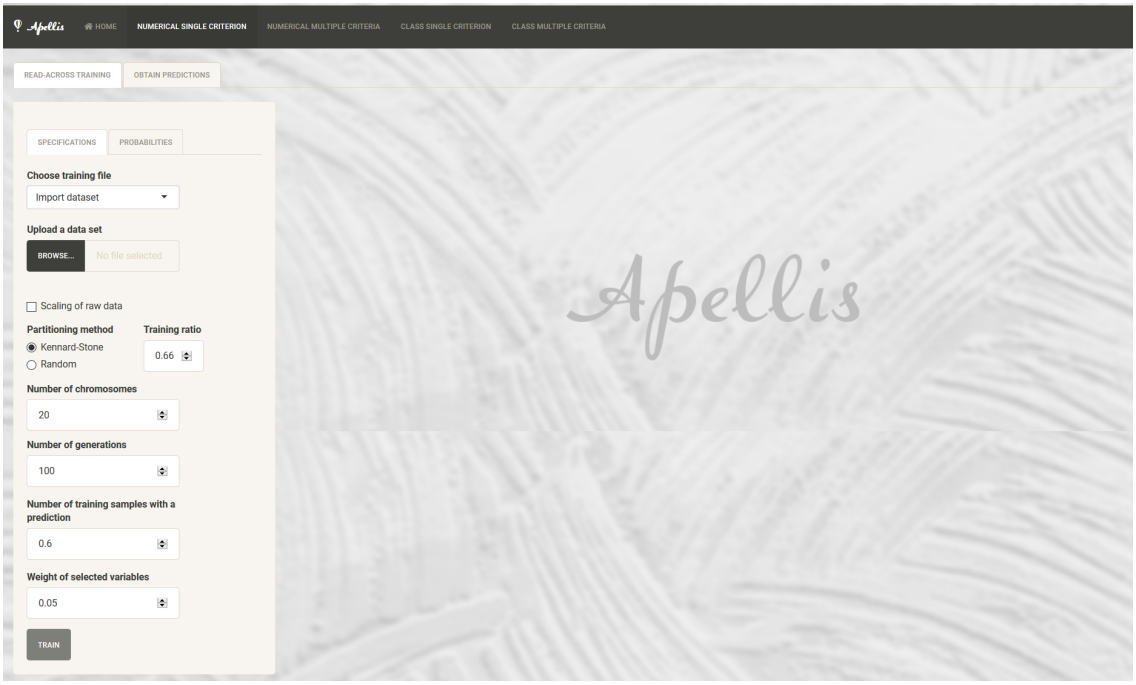

Figure 4: Numerical single criterion tab. On the right part of the interface the specifications that must be tuned are displayed and in the main part the training results are going to be displayed.

4.1 Read-across training

In this section training of a read-across model is performed according to the provided dataset and the set of specifications defined by the user. A sequential workflow, as described previously, is followed in order to find the optimal hypothesis for the endpoint estimation.

A demo dataset is provided coming from Walkey *et al.* (2014)[7] for demonstration reasons. The datasets consists of 84 culture medium incubated gold anionic and cationic nanoparticles (NPs) (diameters of 15, 30 and 60 nm). For each NP there are available 40 physicochemical descriptors, 129 protein corona fingerprints (biological descriptors) and additional measurements of its cell association with human A549 cells (in mL/ g(Mg)). The cell association was transformed into log2 values. The protein corona fingerprints were filtered by Gene Set Variation Analysis and only 63 were considered as statistically significant proteins.[8]

4.1.1 Specifications

Users must upload one .csv file containing the dataset of interest by clicking on the **Browse** button in the **Upload a data set** field. The file must contain the values of available descriptors (in columns separated by commas "," -otherwise an internal error may occur) and the values of the toxicity index (2nd column), which will be predicted by the model. In the 1st column the ENMs names should be listed. Missing values cannot be handled by this approach (Fig. 6). In addition, columns containing the same value in all rows cannot be handled and must be deleted by the input file. An exemplary input file can be seen in Fig. 5.

The user can select if the provided data should be normalized or not, according to Eq. 22 by clicking on **Scaling of raw data**.

$$c_{\mathrm{sc}} = \frac{c_{\mathrm{in}} - min}{max - min} \tag{22}$$

where c_{in}, is the value of the parameter before normalization, min, is the minimum value of the parameter in the

	A	B	C	D	E	F	G	H	I	J	K
1	NP ID	net.c	class	lspri_synth	lspri_serur	lspri_relati	lspri_diff	lspri_rel_c	zav_synth	zav_serum	pdi_synth
2	G15.AC	-5.1839	1	0.18253	0.454404	2.489473	0.271874	1.489473	22.36	57.53	0.084
3	G15.AHT	-1.00854	0	0.45821	0.525747	1.147394	0.067537	0.147394	30.95	90.06	0.399
4	G15.Ala-SH	-5.50439	1	0.223534	0.274761	1.22917	0.051227	0.22917	22.64	44.43	0.147
5	G15.Asn-SH	-5.67669	1	0.27362	0.327264	1.196055	0.053645	0.196055	23.09	37.75	0.15
6	G15.AUT	-1.31567	0	0.365436	0.389573	1.066051	0.024138	0.066051	23.8	55.98	0.326
7	G15.CALNN	-7.13797	1	0.20691	0.265327	1.282332	0.058417	0.282332	25.22	38.8	0.144
8	G15.CIT	-5.41982	1	0.210431	0.292836	1.391602	0.082405	0.391602	18.65	54.03	0.138
9	G15.CTAB	-5.86229	0	0.326142	0.365811	1.121632	0.039669	0.121632	15.6	59.7	0.465
10	G15.DDT@BDHDA	-7.29449	0	0.266579	0.317134	1.189643	0.050555	0.189643	23.15	47.03	0.187
11	G15.DDT@CTAB	-7.59005	0	0.275461	0.324751	1.178939	0.049291	0.178939	20.53	48.4	0.191
12	G15.DDT@DOTAP	-1.12756	0	0.276498	0.297223	1.074954	0.020725	0.074954	28.17	47.34	0.193
13	G15.DDT@ODA	-6.1218	0	0.309989	0.367331	1.184981	0.057342	0.184981	33.6	58.95	0.268
14	G15.DDT@SA	-6.8039	1	0.395907	0.320779	0.810238	-0.07513	-0.18976	82.41	59.93	0.249
15	G15.DDT@SDS	-7.67595	1	0.465011	0.359906	0.773974	-0.1051	-0.22603	27.94	100.13	0.302
16	G15.DTNB	-6.08314	1	0.241281	0.413864	1.715281	0.172583	0.715281	23.58	60.27	0.108
17	G15.F127	-5.36112	1	0.175809	0.244867	1.392798	0.069057	0.392798	42.05	46.63	0.216
18	G15.Gly-SH	-4.97528	1	0.261762	0.402932	1.539304	0.141169	0.539304	77.02	55.39	0.211
19	G15.HDA	-0.27033	0	0.243839	0.275642	1.130427	0.031803	0.130427	28.04	54.89	0.194
20	G15.LA	-5.96398	1	0.236782	0.30131	1.272521	0.064528	0.272521	22.45	48.09	0.23
21	G15.MAA	-6.14203	1	0.281334	0.448587	1.594501	0.167253	0.594501	30.74	60.99	0.273
22	G15.MBA	-5.38142	1	0.293044	0.504305	1.720917	0.211261	0.720917	29.87	67.7	0.321
23	G15.MES	-3.19932	1	0.226324	0.476291	2.104465	0.249967	1.104465	49.22	61.86	0.297
24	G15.Met-SH	-5.9284	1	0.215656	0.28201	1.307686	0.066354	0.307686	23.19	52.42	0.167
25	G15.MHA	-5.73543	1	0.278335	0.389208	1.398344	0.110873	0.398344	25.15	58.99	0.239
26	G15.MHDA	-5.77833	1	0.242059	0.294179	1.215319	0.05212	0.215319	23.12	43.62	0.076
27	G15.MPA	-5.39593	1	0.303644	0.451142	1.485761	0.147498	0.485761	25.73	55.65	0.284
28	G15.MSA	-6.10482	1	0.255577	0.424225	1.659871	0.168648	0.659871	34.33	52	0.203

Figure 5: An exemplary template file for model training (numerical endpoint). The first column contains the ENMs ID, the second the endpoint that is going to be predicted, followed by the rest of descriptors in the rest of the columns.

set, max,is the maximum value of the parameter in the set and c_{sc}, is the normalized value of the parameter.

For validation purposes (formation of training and test sets), users must choose a the partitioning method (Kennard-Stone or random) and the corresponding training:test ratio. The user must initialize some parameters for the evolution of the algorithm presented in Table 4.

Table 4: Apellis' specifications that should be tuned by the users.

Apellis specification	Corresponding GA parameter	Accepted values
Number of chromosomes	*nChrom*	positive even numbers
Number of generations	*generations*	positive numbers
Number of training samples with a prediction	*predFactor*	0-1
Weight of selected variables	wf_{OF}	0-1

4.1.2 Probabilities

In continuation the user must tune a series of operational parameters of the GAs (Fig. 7) that are directly linked to the biological processes of selection, crossover and mutation of genes (Table 5).

Concerning the non-uniform mutation, the lower threshold value is always equal to 0.1, while the upper threshold value is equal to the mean value of the maximum distances between provided samples.

If a dataset is uploaded or the demo dataset is selected, by pressing the **Train** button, the model training starts, according to the described workflow, otherwise the corresponding button remains disabled till all necessary files are provided. During training a process bar is presented, indicating the number of processed generation.

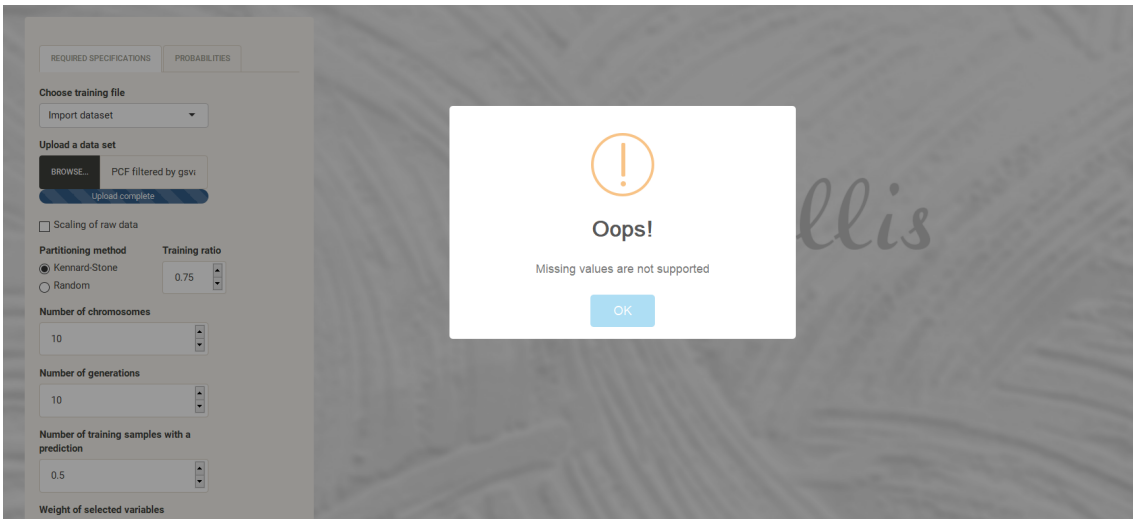

Figure 6: Warning message presented when a file with missing values is uploaded.

Table 5: Probabilities that should be tuned by the users.

Apellis specification	Corresponding GA parameter	Accepted values
Favor variable selection in 1st generation	*initGeneProb*	0-1
Crossover between chromosomes	*crossProb*	0-1
Uniform mutation	*mutProb*	0-1
Non-uniform mutation	*nonUnf*	0-1
Freezing parameter	*bGA*	positive numbers

🚲 It is advised not to leave page nor to refresh it during execution, otherwise training maybe interrupted.

4.1.3 Training results

After the completion of training, a scatter plot containing the predicted and the actual values of all ENMs of the uploaded dataset (training and test sets) is presented. Users can zoom in specific areas of the plot, hover over the depicted points and compare the actual (experimental) and the predicted endpoint values. The diagram can be downloaded in .png format.

The experimental and the predicted endpoint values for the test ENMs with a successful prediction, are also presented in the produced table entitled **Experimental vs. predicted endpoint values for the test set**. The prediction values have the same units as the actual values.

In addition, the app produces two tables one containing the selected variables and one containing information about the trained model (optimized threshold, number of the generation that produced the best solution, the score calculated over the test set, the total number of test samples with a successful prediction, the total number of selected variables, the MSE, the mean absolute error (MAE) and the q^2_{ext} statistic). Finally, the app produces the ENMs' **Neighbors heatmap**, where the neighbors of the test ENMs in the training set are depicted. In this heatmap for each pair of training-test ENMs, a value of 1 (dark color) is allocated if these ENMs are neighbors, or a value of 0 (light color) is allocated if these ENMs are not neighbors. The neighbors heat map actually gives a graphical representation of the ENMs grouping.

Users can download all training results by clicking on **Download training results** in the form of a .zip file, containing .csv files with the selected variables, the neighbors for the test ENMs, the predictions for the test ENMs and all necessary statistics and model information.

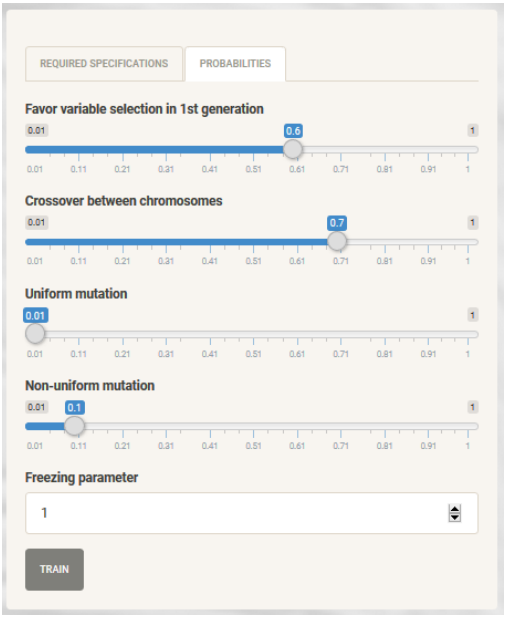

Figure 7: Read-across training sub-tab for tuning the probabilities presented in Table 5. The probabilities' tab is the same in training using single or multiple criteria.

Users can also name their model by typing in **Model title** and download the trained model by clicking on **Download model** for future use in the following tab (**Obtain predictions**) without any need for constant training with the same data. An example of training is presented in Fig. 8.

4.2 Obtain predictions

This section of the application can be used right after the read-across training or after the input of an already trained model by **Apellis** (Fig. 9). In that way, the toxicity index of a data set of untested ENMs can be predicted, when all the descriptors that were used in training are known values.

If a model is already available from a previous training, the users can click on **Upload model** and import an adequate model file by clicking on **Browse** button of the **Upload an adequate model file** field. When the model is uploaded a **Template input file** can be downloaded in order to be filled-in by the users with the necessary descriptor/variables values that will be used for the endpoint prediction. The necessary variables are also presented, as well as the endpoint that will be predicted, the model title and the q^2_{ext} statistic from external validation.

This part can be also used right after the training of the model. In that case the **Upload model** field remains disabled. The user must upload a .csv file in the application containing the descriptor values for the untested ENMs dataset, either according to the template file (in case that a model is uploaded) or according to the training file (omitting the 2nd column). It is necessary to provide values only for the selected descriptors, however the columns of the non-selected variables must not be deleted; it is advisable to be filed with miscellaneous values.

By clicking on **Predict** button, the prediction process begins. The analysis produces a table that contains the predicted value of toxicity index for all the provided ENMs, and the ENMs **Neighbors heatmap**. All results can be downloaded in .zip format by clicking on **Download prediction results**. An example of model use can be found in Fig. 10.

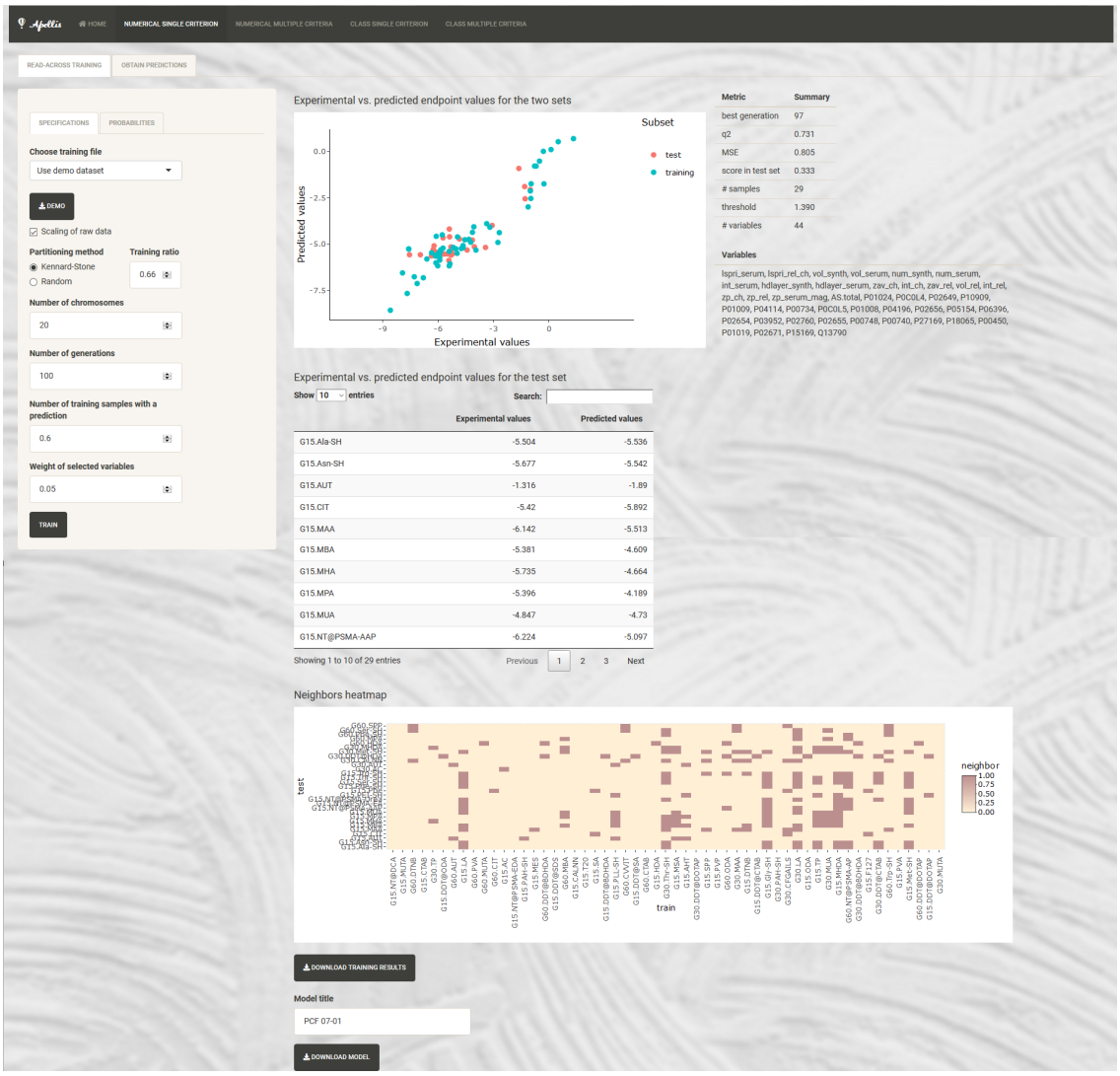

Figure 8: Numerical single criterion tab training results.

5 Numerical multiple criteria

This part (Fig. 11) can be used for training and using a read-across model for the prediction of toxicity endpoints and other properties of ENMs, when two types of descriptors are available (e.g. physicochemical, image, biological, theoretical etc. descriptors).

5.1 Read-across training

In this section training of a read-across model is performed according to the provided dataset and the set of specifications defined by the user. The same demo dataset as in **Numerical single criterion** is provided. [7], [8]

5.1.1 Specifications

Users must upload two .csv files containing the dataset of interest by clicking on the **Browse** button in the **Upload a data set** type 1 or 2 fields. The first file (type 1) must contain the values of available type 1 descriptors (in

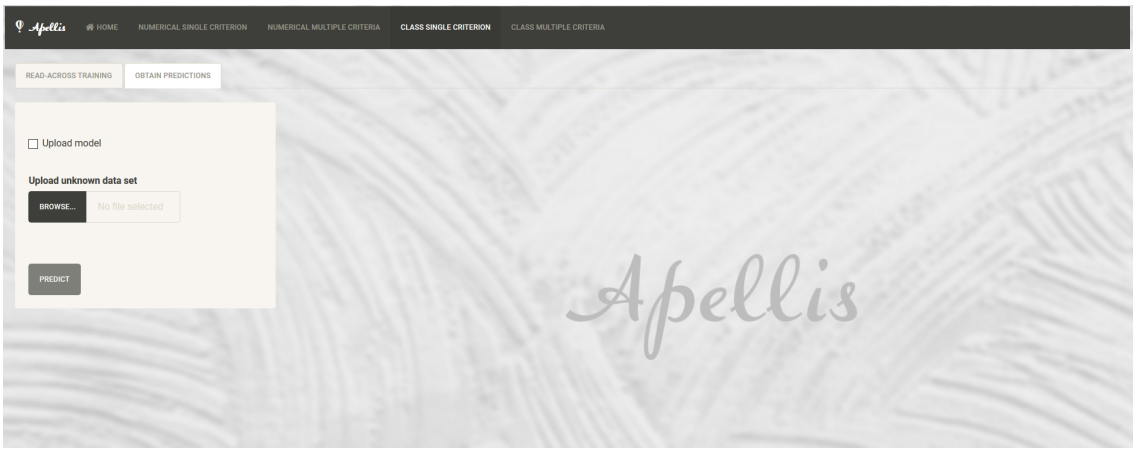

Figure 9: Numerical single criterion tab for model use. This interface is similar for the numerical multiple criteria, class single criterion and class multiple criteria tabs.

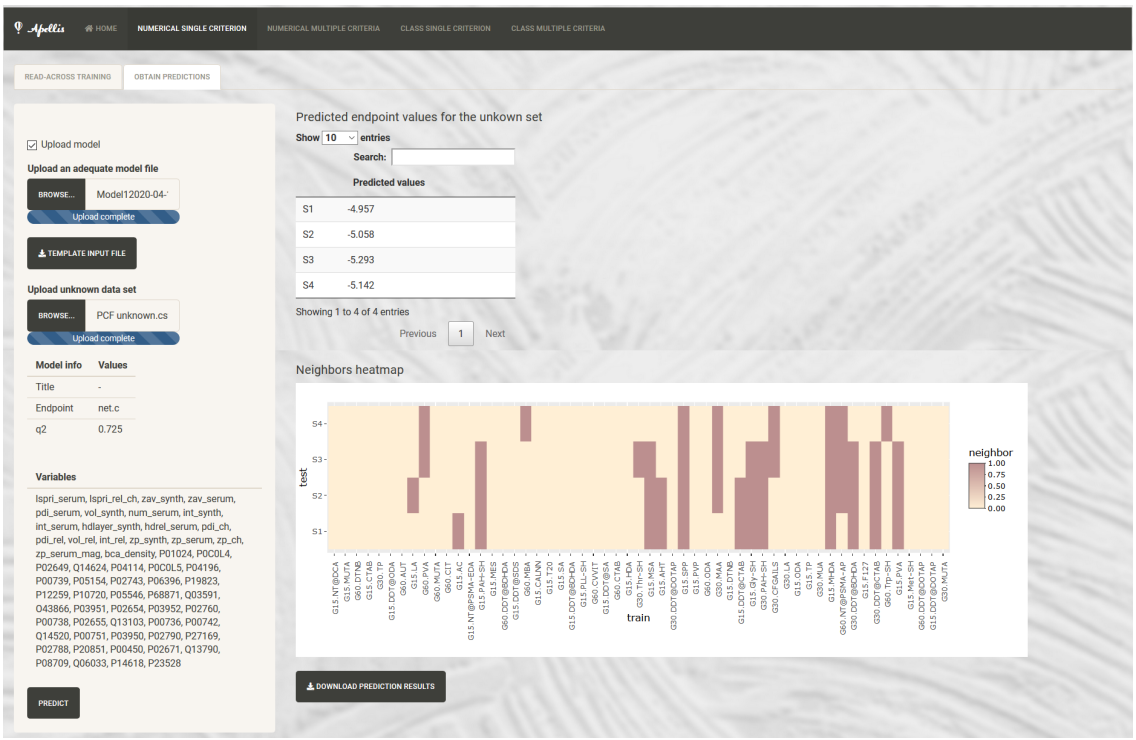

Figure 10: Numerical single criterion tab for model use. In this example an already trained model is uploaded to the app. Its details can be found in the right side of the interface. The predictions for an artificial dataset of gold NPs are presented on the left side of the interface.

columns) and the values of the toxicity index (2nd column), which will be predicted by the model. In the 1st column the ENMs names should be listed. The second file (type 2) must contain the values of available type 2 descriptors (in columns). In the 1st column the ENMs names should be listed in the same order as in the type 1 file. Missing values cannot be handled by this approach. In addition, columns containing the same value in all rows cannot be handled and must be deleted by the input file. An example of input files can be seen in Fig. 12.

The user can select if the provided data should be normalized or not, according to Eq. 22 by clicking on **Scaling of raw data**.

For validation purposes (formation of training and test sets), users must choose a the partitioning method (Kennard-

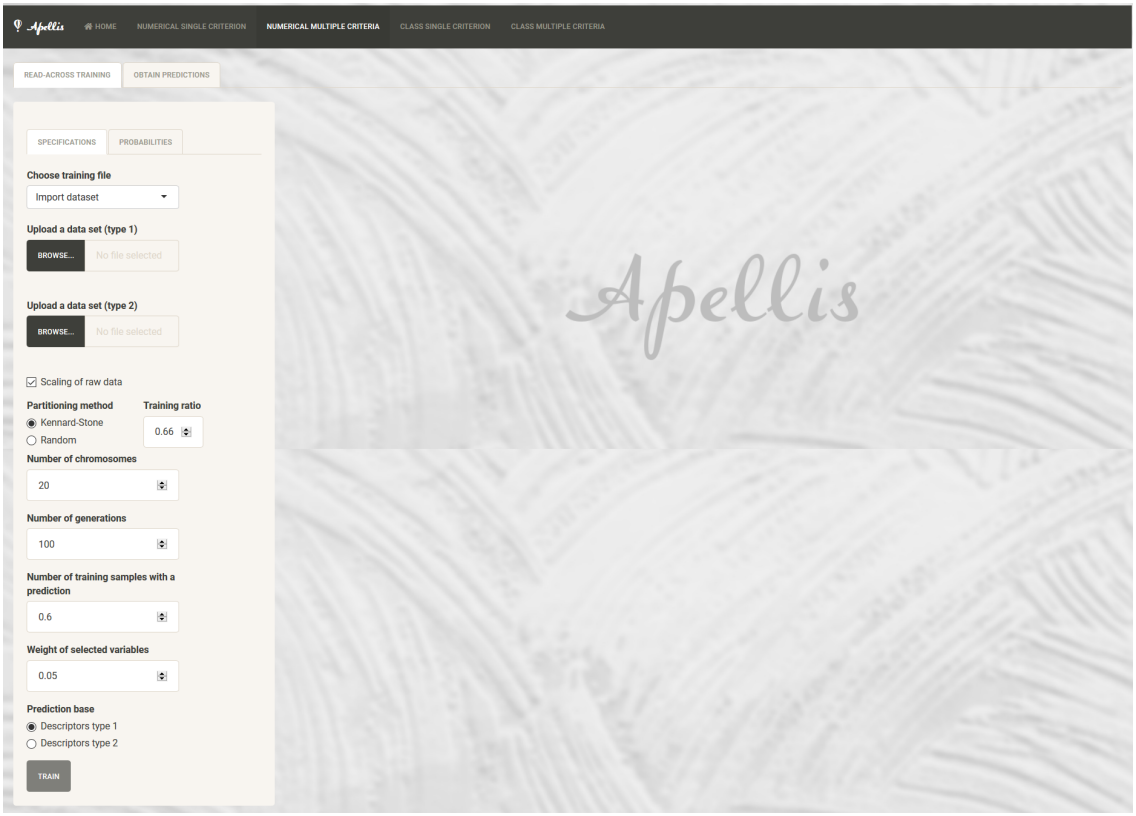

Figure 11: Numerical multiple criterion tab. On the right part of the interface the specifications that must be tuned are displayed and in the main part the training results are going to be displayed.

Stone or random) and the corresponding training:test ratio. The users must initialize some parameters for the evolution of the algorithm presented in Table 4. Apart from the parameters presented in Table 4, the users must define the **Prediction base**; namely the weighting factors type (either calculated from distances computed using descriptors type 1 either type 2) that will be used in predictions (see Eq. 6).

5.1.2 Probabilities

As explained before, the user must tune a series of operational parameters of the GAs that are directly linked to the biological processes of selection, crossover and mutation of genes (Table 5).

Concerning the non-uniform mutation, the lower threshold value is always equal to 0.1, while the upper threshold value is equal to the mean value of the maximum distances between provided samples.

If a dataset is uploaded or the demo dataset is selected, by pressing the **Train** button, the model training starts, according to the described workflow, otherwise the corresponding button remains disabled till all necessary files are provided. During training a process bar is presented, indicating the number of processed generation.

5.1.3 Training results

After the completion of training, a scatter plot containing the predicted and the actual values of all ENMs of the uploaded dataset (training and test sets) is presented. The diagram can be downloaded in .png format.

The experimental and the predicted endpoint values for the test ENMs with a successful prediction, are also presented in the produced table entitled **Experimental vs. predicted endpoint values for the test set**. The prediction values have the same units as the actual values.

	A	B	C	D	E	F
1	NP ID	net.c	P01024	P0C0L4	P02649	P10909
2	G15.AC	-5.18390005	0.073492144	0.027369488	0.085656361	0.012671059
3	G15.AHT	-1.00853711	0.021656535	0.055091185	0.001899696	0.005319149
4	G15.Ala-SH	-5.504386695	0.08105939	0.017656501	0.107544141	0.00882825
5	G15.Asn-SH	-5.676687582	0.084718923	0.015835313	0.102137767	0.005542359
6	G15.AUT	-1.315665982	0.024524313	0.054968288	0.012262156	0.008033827
7	G15.CALNN	-7.13796526	0.043650794	0.034391534	0.010582011	0.007936508
8	G15.CIT	-5.419815916	0.127461707	0.038840263	0.012035011	0.007658643
9	G15.CTAB	-5.862286645	0.091865358	0.074333801	0.01542777	0.010518934
10	G15.DDT@BDHDA	-7.294490912	0.103219697	0.026515152	0.015151515	0.023674242
11	G15.DDT@CTAB	-7.590049746	0.092989986	0.026466381	0.015736767	0.04434907
12	G15.DDT@DOTAP	-1.127557324	0.043281654	0.033591731	0.004521964	0.010981912
13	G15.DDT@ODA	-6.121800441	0.044052863	0.008810573	0.008810573	0.070484581
14	G15.DDT@SA	-6.803896602	0.125886525	0.01891253	0.046099291	0.088652482
15	G15.DDT@SDS	-7.675949819	0.107542942	0.044062733	0.023898432	0.035847647

	A	B	C	D	E	F
1	NP ID	class	lspri_synth	lspri_serum	lspri_relative	lspri_diff
2	G15.AC	1	0.182530253	0.454404195	2.48947332	0.271873942
3	G15.AHT	0	0.458209658	0.525747071	1.147394127	0.067537412
4	G15.Ala-SH	1	0.223533915	0.274761252	1.22917031	0.051227337
5	G15.Asn-SH	1	0.273619886	0.327264445	1.196055046	0.053644559
6	G15.AUT	0	0.365435833	0.389573359	1.066051338	0.024137526
7	G15.CALNN	1	0.206909736	0.26532699	1.28233207	0.058417254
8	G15.CIT	1	0.210430919	0.292836119	1.391602152	0.082405201
9	G15.CTAB	0	0.326141556	0.365810851	1.121632137	0.039669294
10	G15.DDT@BDHDA	0	0.266578823	0.317133738	1.189643401	0.050554915
11	G15.DDT@CTAB	0	0.275460611	0.32475128	1.17893908	0.049290668
12	G15.DDT@DOTAP	0	0.27649778	0.297222531	1.074954492	0.020724751
13	G15.DDT@ODA	0	0.309989179	0.367331174	1.184980632	0.057341994
14	G15.DDT@SA	1	0.395907163	0.320778997	0.810237923	-0.075128165
15	G15.DDT@SDS	1	0.465010844	0.359906092	0.773973547	-0.105104752

Figure 12: Two exemplary template files (types 1 and 2) for numerical model training. Type 1 file (left image): The first column contains the ENMs ID, the second the endpoint that is going to be predicted, followed by the rest of descriptors in the rest of the columns.Type 2 file (right image): The first column contains the ENMs ID, followed by the rest of descriptors in the rest of the columns.

In addition, the app produces two tables one containing the selected variables and one containing information about the trained model. Finally, the app produces the ENMs' **Neighbors heatmap**, where the neighbors of the test ENMs in the training set are depicted.

Users can download all training results by clicking on **Download training results** in the form of a .zip file, containing .csv files with the selected variables, the neighbors for the test ENMs, the predictions for the test ENMs and all necessary statistics and model information.

Users can also name their model by typing in **Model title** and download the trained model by clicking on **Download model** for future use in the following tab (**Obtain predictions**) without any need for constant training with the same data. The results are similar to the training results of **Numerical Single Criterion** tab (Fig. 8).

5.2 Obtain predictions

This section of the application can be used right after the read-across training or after the input of an already trained model by **Apellis**. In that way, the toxicity index of a data set of untested ENMs can be predicted, when all the descriptors that were used in training are known values. The interface for model use is similar to the one of the **Numerical single Criterion** (Fig. 9).

If a model is already available from a previous training using **Numerical multiple criteria**, the users can click on **Upload model** and import an adequate model file by clicking on **Browse** button of the **Upload an adequate model file** field. When the model is uploaded a **Template input file** can be downloaded in order to be filled-in by the users with the necessary descriptor/variables values that will be used for the endpoint prediction. The necessary variables are also presented, as well as the endpoint that will be predicted, the model title and the q^2_{ext} statistic from external validation.

This part can be also used right after the training of the model. In that case the **Upload model** field remains disabled. The user must upload a .csv file in the application containing the descriptor values for the untested ENMs dataset, either according to the template file (in case that a model is uploaded) or according to the training file (omitting the 2nd column). It is necessary to provide values only for the selected descriptors, however the columns of the non-selected variables must not be deleted; it is advisable to be filed with miscellaneous values.

By clicking on **Predict** button, the prediction process begins. The analysis produces a table that contains the predicted value of toxicity index for all the provided ENMs, and the ENMs **Neighbors heatmap**. All results can be downloaded in .zip format by clicking on **Download prediction results**.

6 Class single criterion

Similarly to numerical endpoints, users can train and use a categorical-endpoint read-across model from the adequate tabs. The **Class single criterion** part can be used for training and using a read-across model for the prediction of toxicity class of ENMs, when only one type of descriptors is available (or users aspire to treat the descriptors as one category).

6.1 Read-across training

In this section training of a read-across model is performed according to the provided dataset and the set of specifications defined by the user. The interface is the same as for numerical endpoints training. A demo dataset is provided coming from Liu *et al.* (2013) [9] for demonstration reasons. The datasets consists of 23 metal oxide-cored ENMs. For each ENM there are 24 available descriptors and its toxicity class (TRUE/FALSE) according to the toxicity profiles consisting of seven different assays for human bronchial epithelial (BEAS-2B) and murine myeloid (RAW 264.7) cells, over a concentration range of 0.39–100 mg/L and exposure time up to 24 hrs.

6.1.1 Specifications

The **Specifications** part is similar to the **Numerical single criterion** part (Fig. 4). Users must upload one .csv file containing the dataset of interest by clicking on the **Browse** button in the **Upload a data set** field. The file must contain the values of available descriptors (in columns separated by commas ",") and the class (TRUE/FALSE) of the toxicity index (2nd column), which will be predicted by the model. In the 1st column the ENMs names should be listed. Missing values cannot be handled by this approach. In addition, columns containing the same value in all rows cannot be handled and must be deleted by the input file. An exemplary input file can be seen in Fig. 13.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	NP	Toxicity	dp	EC	EV	EAmz	χMeO	ΔHsub	ΔHIE	ΔHsf	ΔHLat	ΔHIE	Z2/r	IEP	ZP
2	ZnO	TRUE	22.6	-3.891	-7.198	7.546	5.674	1.351	28.71	-3.608	42.928	9.394	0.0667	9.6	28.8
3	CuO	TRUE	12.8	-5.174	-6.515	7.719	5.874	3.497	31.515	-1.609	42.856	7.726	0.0548	7.9	7.6
4	Mn2O3	TRUE	51.5	-4.647	-7.635	11.709	5.919	2.936	59.677	-9.917	156.975	7.434	0.1552	3.7	-46.1
5	CoO	TRUE	71.8	-4.424	-6.832	9.454	5.735	4.422	29.387	-2.476	39.767	7.881	0.0615	9.2	21.6
6	Co3O4	TRUE	10	-4.593	-7.025	10.755	5.927	4.422	46.137	-9.38	99.573	7.881	0.1329	9.4	24.6
7	Cr2O3	TRUE	193	-4.439	-7.524	13.92	5.858	4.12	58.331	-11.717	158.322	6.767	0.1452	5.3	-32.6
8	Ni2O3	TRUE	140.6	-4.309	-7.688	11.709	6.052	4.458	65.455	-5.073	164.157	7.639	0.1607	8.3	32.2
9	Gd2O3	FALSE	43.8	-2.825	-8.102	16.782	5.499	4.12	42.989	-18.82	134.692	6.15	0.0957	8	6.5
10	In2O3	FALSE	59.6	-3.632	-7.322	11.188	5.583	2.518	55.204	-9.606	144.351	5.786	0.1125	9.2	61.9
11	CeO2	FALSE	18.3	-3.803	-7.45	20.121	5.65	4.354	77.697	-11.284	99.775	5.539	0.1649	7.8	21.4
12	SiO2	FALSE	13.5	-2.018	-11.118	18.734	6.19	4.664	107.795	-9.41	136.029	8.151	0.6154	1	-31.8
13	Al2O3	FALSE	14.7	-1.515	-9.815	15.872	5.665	3.429	56.691	-17.345	164.955	5.985	0.1667	7.4	0
14	Y2O3	FALSE	32.7	-2.352	-8.201	17.433	5.406	4.402	43.362	-19.748	131.676	6.217	0.1	9.6	42.7
15	SnO2	FALSE	62.4	-4.013	-8.013	14.397	5.812	3.122	96.334	-5.986	122.369	7.344	0.2319	4	-38.8
16	TiO2	FALSE	12.6	-4.161	-7.491	19.775	5.767	4.902	96.063	-9.779	125.924	6.828	0.2623	6.4	-19.4
17	ZrO2	FALSE	40.1	-3.192	-8.233	22.723	5.618	6.322	83.379	-11.252	115.954	6.634	0.1905	5.8	-12.8
18	Fe2O3	FALSE	12.3	-4.993	-6.987	12.489	5.978	4.306	59.047	-8.512	148.3	7.903	0.1636	7.2	-2.1
19	Sb2O3	FALSE	11.8	-3.645	-8.138	10.408	5.514	2.74	53.278	-7.346	142.071	8.608	0.1184	1	-35.3
20	HfO2	FALSE	28.4	-2.956	-8.371	23.938	5.705	6.409	84.863	-1.17	104.812	6.825	0.1928	8.1	33.5
21	WO3	FALSE	16.6	-5.532	-8.586	24.978	6.64	8.82	213.421	-8.734	250.324	7.864	0.6	0.3	-61.3
22	Yb2O3	FALSE	61.7	-2.831	-7.933	15.091	5.429	1.613	45.092	-18.807	138.672	6.254	0.0909	8.2	9.9
23	La2O3	FALSE	24.6	-2.38	-8.147	17.433	5.378	4.467	40.28	-18.668	129.054	5.577	0.0776	9.4	54.3
24	NiO	FALSE	13.1	-3.57	-7.445	9.454	5.744	4.458	30.266	-2.494	40.503	7.639	0.058	11.4	27.6

Figure 13: An exemplary template file for model training (categorical endpoint). The first column contains the ENMs ID, the second the class (TRUE/FALSE) that is going to be predicted, followed by the rest of descriptors in the rest of the columns.

Again, the user can select if the provided data should be normalized or not, according to Eq. 22 by clicking on **Scaling of raw data**, the partitioning method (Kennard-Stone or random) for external validation and the corresponding training:test ratio. The user must also initialize some parameters for the evolution of the algorithm as before (Table 4).

6.1.2 Probabilities

The operational parameters of the GAsfor the selection, crossover and mutation of genes must also be tuned (Table 5, Fig. 7).

Concerning the non-uniform mutation, the lower threshold value is always equal to 0.1, while the upper threshold value is equal to the mean value of the maximum distances between provided samples.

If a dataset is uploaded or the demo dataset is selected, by pressing the **Train** button, the model training starts, according to the described workflow, otherwise the corresponding button remains disabled till all necessary files are provided. During training a process bar is presented, indicating the number of processed generation.

6.1.3 Training results

After the completion of training, a confusion matrix is presented depicting the validation results (TP, TN, FP, FN frequencies). The experimental and the predicted endpoint values for the test ENMs with a successful prediction, are also presented in the produced table entitled **Experimental vs. predicted endpoint class for the test set**.

In addition, the app produces two tables one containing the selected variables and one containing information about the trained model (optimized threshold, number of the generation that produced the best solution, the score calculated over the test ENMs, the total number of test samples with a successful prediction, the total number of selected variables and the accuracy, sensitivity and specificity statistics, and MCC). Finally, the app produces the ENMs' **Neighbors heatmap**, where the neighbors of the test ENMs in the training set are depicted. In this heatmap for each pair of training-test ENMs, a value of 1 (dark color) is allocated if these ENMs are neighbors, or a value of 0 (light color) is allocated if these ENMs are not neighbors.

Users can download all training results by clicking on **Download training results** in the form of a .zip file, containing .csv files with the selected variables, the neighbors for the test ENMs, the predictions for the test ENMs and all necessary statistics and model information.

Users can also name their model by typing in **Model title** and download the trained model by clicking on **Download model** for future use in the following tab (**Obtain predictions**) without any need for constant training with the same data. An example of training is presented in Fig. 14.

6.2 Obtain predictions

This section of the application can be used right after the read-across training or after the input of an already trained model by **Apellis** (the interface is similar to the tab presented in Fig. 9).

If a model is already available from a previous training, the users can click on **Upload model** and import an adequate model file by clicking on **Browse** button of the **Upload an adequate model file** field. When the model is uploaded a **Template input file** can be downloaded in order to be filled-in by the users with the necessary descriptor/variables values that will be used for the endpoint prediction. The necessary variables are also presented, as well as the endpoint that will be predicted, the model title and the accuracy statistic from external validation.

This part can be also used right after the training of the model. In that case the **Upload model** field remains disabled. The user must upload a .csv file in the application containing the descriptor values for the untested ENMs dataset, either according to the template file (in case that a model is uploaded) or according to the training file (omitting the 2nd column). It is necessary to provide values only for the selected descriptors, however the columns of the non-selected variables must not be deleted; it is advisable to be filed with miscellaneous values.

By clicking on **Predict** button, the prediction process begins. The analysis produces a table that contains the predicted value of toxicity index for all the provided ENMs, and the ENMs **Neighbors heatmap**. All results can be downloaded in .zip format by clicking on **Download prediction results**. An example of model use can be found in Fig. 15.

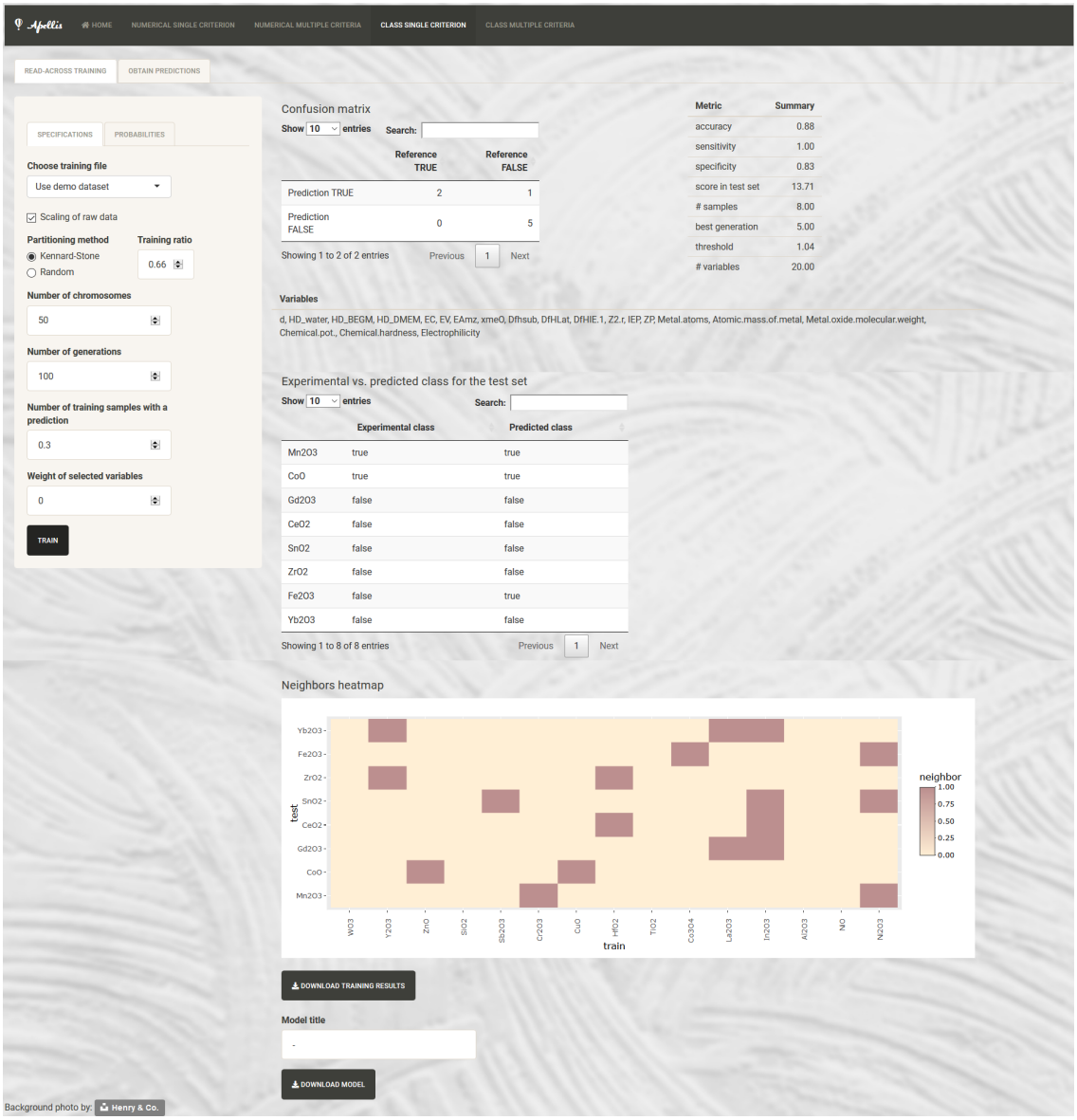

Figure 14: Class single criterion tab training results, including the confusion matrix for the test set, the corresponding accuracy metrics, the predicted class for each test sample and the neighbors heatmap.

7 Class multiple criteria

This part can be used for training and using a read-across model for the prediction of toxicity endpoints and other properties of ENMs, when two types of descriptors are available (e.g. physicochemical, image, biological, theoretical etc. descriptors).

7.1 Read-across training

In this section training of a read-across model is performed according to the provided dataset and the set of specifications defined by the user.

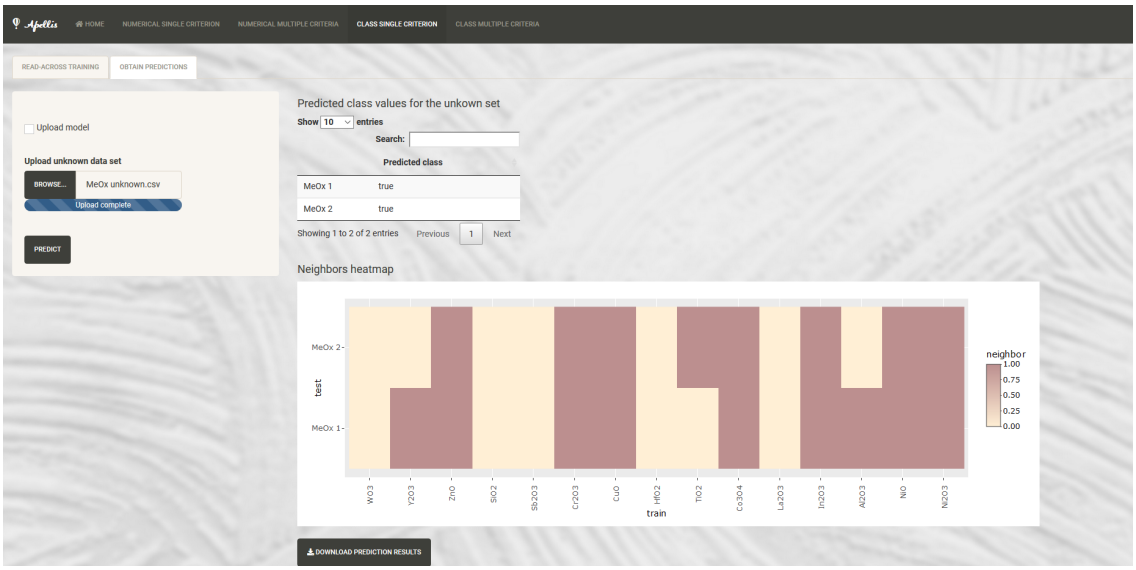

Figure 15: Single criterion tab for model use. In this example predictions for a set of samples with unknown class endpoints, are acquired directly after the model train. The predictions for an artificial dataset of metal oxide ENMs are presented on the left side of the interface.

7.1.1 Specifications

Users must upload two .csv files containing the dataset of interest by clicking on the **Browse** button in the **Upload a data set** type 1 or 2 fields. The first file (type 1) must contain the values of available type 1 descriptors (in columns) and the values of the toxicity (TRUE/FALSE) index (2nd column), which will be predicted by the model. In the 1st column the ENMs names should be listed. The second file (type 2) must contain the values of available type 2 descriptors (in columns). In the 1st column the ENMs names should be listed in the same order as in the type 1 file. Missing values cannot be handled by this approach. In addition, columns containing the same value in all rows cannot be handled and must be deleted by the input file.

The user can select if the provided data should be normalized or not, according to Eq. 22 by clicking on **Scaling of raw data**.

For validation purposes (formation of training and test sets), users must choose a the partitioning method (Kennard-Stone or random) and the corresponding training:test ratio. The users must initialize some parameters for the evolution of the algorithm presented in Table 4. Apart from the parameters presented in Table 4, the users must define the **Prediction base**; namely the weighting factors type (either calculated from distances computed using descriptors type 1 either type 2) that will be used in predictions (see Eq. 6).

7.1.2 Probabilities

As explained before, the user must tune a series of operational parameters of the GAs that are directly linked to the biological processes of selection, crossover and mutation of genes (Table 5, Fig. 7).

Concerning the non-uniform mutation, the lower threshold value is always equal to 0.1, while the upper threshold value is equal to the mean value of the maximum distances between provided samples.

If a dataset is uploaded by pressing the **Train** button, the model training starts, according to the described workflow, otherwise the corresponding button remains disabled till all necessary files are provided. During training a process bar is presented, indicating the number of processed generation.

7.1.3 Training results

After the completion of training, a confusion matrix is presented depicting the validation results (TP, TN, FP, FN frequencies). The experimental and the predicted endpoint values for the test ENMs with a successful prediction, are also presented in the produced table entitled **Experimental vs. predicted endpoint values for the test set**.

In addition, the app produces two tables one containing the selected variables and one containing information about the trained model. Finally, the app produces the ENMs' **Neighbors heatmap**, where the neighbors of the test ENMs in the training set are depicted. The training results are similar to the **Class single criterion** tab (Fig. 14).

Users can download all training results by clicking on **Download training results** in the form of a .zip file, containing .csv files with the selected variables, the neighbors for the test ENMs, the predictions for the test ENMs and all necessary statistics and model information.

Users can also name their model by typing in **Model title** and download the trained model by clicking on **Download model** for future use in the following tab (**Obtain predictions**) without any need for constant training with the same data.

7.2 Obtain predictions

This section of the application can be used right after the read-across training or after the input of an already trained model by **Apellis**. If a model is already available from a previous training using **Class multiple criteria**, the users can click on **Upload model** and import an adequate model file by clicking on **Browse** button of the **Upload an adequate model file** field. When the model is uploaded a **Template input file** can be downloaded in order to be filled-in by the users with the necessary descriptor/variables values that will be used for the endpoint prediction. The necessary variables are also presented, as well as the endpoint that will be predicted, the model title and the accuracy statistic from external validation.

This part can be also used right after the training of the model. In that case the **Upload model** field remains disabled. The user must upload a .csv file in the application containing the descriptor values for the untested ENMs dataset, either according to the template file (in case that a model is uploaded) or according to the training file (omitting the 2nd column). It is necessary to provide values only for the selected descriptors, however the columns of the non-selected variables must not be deleted; it is advisable to be filed with miscellaneous values.

By clicking on **Predict** button, the prediction process begins. The analysis produces a table that contains the predicted value of toxicity index for all the provided ENMs, and the ENMs **Neighbors heatmap**. All results can be downloaded in .zip format by clicking on **Download prediction results**.

8 Abbreviations

- ENM, engineered nanomaterial
- GA, genetic algorithm
- MAE, mean absolute error
- MCC, Matthew's correlation coefficient
- MSE, mean squared error

9 Disclaimer

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see https://www.gnu.org/licenses/.

10 Contact information

dimitra.varsou@gmail.com

DemetraDanae

Video tutorial

Unit of Process Control and Informatics, School of Chemical Engineering, NTUA, GR

Acknowledgments

This research work was supported by the Hellenic Foundation for Research and Innovation (HFRI) under the HFRI PhD Fellowship grant (Fellowship Number: 637).

References

[1] ECHA, *Read Across Assessment Framework, Appendix R.6-1 for nanomaterials applicable to the Guidance on QSARs and Grouping of Chemicals. Guidance on information requirements and chemical safety assessment*, 2017, Available online in: https://echa.europa.eu/documents/10162/23036412/appendix_r6_nanomaterials_en.pdf

[2] D. D. Varsou, A. Afantitis, G. Melagraki, H. Sarimveis, *Read-across predictions of nanoparticle hazard endpoints: a mathematical optimization approach*, Nanoscale Advances, 1(9): 3485-3498, 2019, Available online in: https://pubs.rsc.org/en/content/articlehtml/2019/na/c9na00242a

[3] M. Daszykowski, B. Walczak and D. Massart, *Representative subset selection*, Analytica Chimica Acta, 468: 91–103, 2002, Available online in: https://www.sciencedirect.com/science/article/pii/S0003267002006517?via%3Dihub

[4] A. Alexandridis, P. Patrinos, H. Sarimveis, G. Tsekouras, *A two-stage evolutionary algorithm for variable selection in the development of RBF neural network models*, Chemometrics and intelligent laboratory systems,75:2, 149-162, 2005

[5] A. Tropsha, *Best practices for QSAR model development, validation, and exploitation*, Molecular informatics, 29(6-7): 476–488, 2010

[6] A. Tropsha, P. Gramatica, V. K. Gombar, *The importance of being earnest: validation is the absolute essential for successful application and interpretation of QSPR models*,QSAR & Combinatorial Science, 22(1): 69-77, 2003

[7] C. D. Walkey, J. B. Olsen, F. Song, R. Liu, H. Guo, W. Olsen, Y. Cohen, A. Emili, W. C. W. Chan, *Protein Corona Fingerprinting Predicts the Cell Association of Gold Nanoparticles*, ACS Nano, 8 (3), 2439–2455, 2014, Available online in: https://pubs.acs.org/doi/abs/10.1021/nn406018q

[8] D. D. Varsou, G. Tsiliki, P. Nymark, P. Kohonen, R. Grafström, H. Sarimveis, *toxFlow: a web-based application for read-across toxicity prediction using omics and physicochemical data*, Journal of Chemical Information and Modeling, 58(3): 543-549, 2017, Available online in: https://pubs.acs.org/doi/10.1021/acs.jcim.7b00160

[9] R. Liu, H. Yuan Zhang, Z. Xia Ji, R. Rallo, T. Xia, C. Hyun Chang, A. Nel,Y. Cohen, *Development of structure–activity relationship for metal oxide nanoparticles*, Nanoscale, 5: 5644–5653, 2013, Available online in: https://pubs.rsc.org/en/content/articlelanding/2013/NR/c3nr01533e#!divAbstract