Machine Learning for Toxicological Testing

Manuel Leone, Gabriele Macchi, Marco Vicentini
Mentor: Marco Baity-Jesi
Department of Computer Science, EPFL Lausanne, Switzerland

***Abstract*—Toxicological databases are large compilations of data from environmental toxicity studies. In this paper we got data from the Ecotoxicology Database (ECOTOX), which provides information about the adverse effects of single chemicals stressors to some ecologically relevant species. In our work we will focus on a subset of this ECOTOX database and we will use k-Nearest Neighbors (k-NN) and Matrix Factorization algorithms to predict the effect of untested chemicals on tested species and vice versa. We managed to perform an accuracy of 0.9 and 0.7 on a binary classification and a 5-multiclass classification respectively, which are satisfactory results given the high variability of the results across repeated experiments.**

I. Introduction

This project aims to predict the **sensibility** of a particular organism concerning a given chemical. Specifically, by setting thresholds on the concentration needed to kill a half of the population (called LC50 or EC50), we want to give a *"score"* to each pair species-chemical (under certain testing condition) and predict this score. In the first instance, the project itself aims to predict if 1 mg/L of a certain chemical is enough to have the LC50 effect (**binary classification**). In the second part we implement a **multiclass prediction** in order not to limit on a specific density of the chemical but on classes of density ranges. The need for Machine Learning (ML) techniques on this problem it's given by the disorder among the different experiments, with a high level of **subjectivity** that can be solved by using a Machine Learning algorithm. We got data from the **ECOTOX** [1] dataset, containing information about chemical toxicity data for aquatic life, terrestrial plants and wildlife, including more than 952,000 test records covering 12,900 aquatic and terrestrial species and 11,750 chemicals. In this work, we are focusing only on the water life system. For the purpose of the project, we decided to focus on two main ML techniques: **k-NN** classification (Section III) and **Matrix Factorization** (Section IV).

II. Data Analysis & Feature Engineering

Implementing a good ML algorithm for this particular case requires data analysis skills and understanding of the chemical field. Merging these two is fundamental to bargain with the ECOTOX dataset. A preliminary phase of features selection and engineering is a cardinal step toward a final good result. At the end of this section, the dataset will be split in a **70% train set** and a **30% test set**, and these two will be used to evaluate the performance of both of our models.

A. Preliminary filtering

The very first step of the data analysis process is to filter the dataset. To do so, we select only the **LC50** and **EC50** final effects (defined as *endpoints* by ECOTOX), inside the **mortality** group. This means considering the mortality of 50% of the species. In addition, the *embryos* are discarded (as they are not relevant for the analysis) and only a subset of the total species is considered: the **Fish Group**.
At the end of this preliminary phase the dataset counts of **64746** rows and **272** possible features.

B. Features extraction

Even though the set of possible features is huge, the majority of them present many null values: as reported in Figure 1, only 30 columns have less than 10% of null values. Also, many features are not suitable for the purpose of this project. For these two main reasons, we put our efforts on **4** main categories of features, trying to gather the best from each of them.

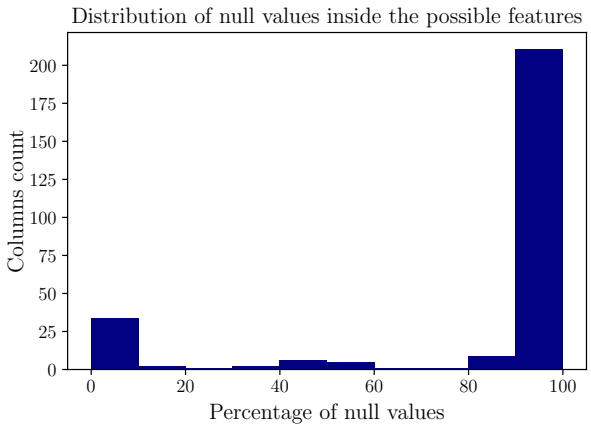

Fig. 1. Count of percentages of null values inside the ECOTOX dataset.

1) CAS Registry Number and Species characteristics

The *CAS* and the *species name* uniquely identify the chemical used for testing and the species on which the test was performed. Further, from the CAS we can extract the chemical properties of the compounds, providing more features (see discussion on the SIMILES identifiers in Sec. II-B4).

The species division through the organisms kingdom defines categorical features for each species. There are 4 categories inside the dataset: *class*, *tax order*, *family* and *genus*.

2) Test duration

The study duration can be a strong feature for our ML algorithm. We proceeded to analyze all of them in order to find the best combination:

- *Exposure Duration*: 7 out of 10 possible features have 99% of null values so are impossible to use. While analyzing this category we found a useful feature, *exposure*

type: this column contains only only **8.22%** nulls, with 21 possible outputs.
- *Observation Duration*: The *observation duration* and the correlated *observation duration unit* present only **1.86%** and **0.3%** of null values respectively.

Therefore the cited features are selected and we proceeded to impute them.

The outcomes of the columns of *exposure type* have some values labeled as **NR**, this implies a water environment, so they all are labeled as AQUA. The possible outcomes of *observation duration* are related to the *unit* chosen to measure time in the single experiment, so everything is converted into hours. The duration distribution shows that **80%** of the dataset is covered by **4** possible duration: **24, 48, 72** and **96** hours. For this reason we keep only these 4, basically turning the feature in a categorical one.

3) Concentration

The chemical's concentration is used to label the different datapoints. In the ECOTOX dataset there are several different measures related to this characteristic (28 in total). However, only 3 of them have a total number of NaN values less or equal 50%:

1) *Concentration value*: this column is the most important, because it's the **label** for our ML model.
2) *Concentration unit*: is directly connected to the label defined the concentration. The **first 10** units cover the 97% of the total. For this reason, only them are kept for future analysis.
3) *Concentration type*: good categorical feature with 2.7% of missing values (which are removed) and 8 possible outputs.

We imputed the concentration values to correctly define our labels. The 10 units are converted to just one: **mg/L**. The missing values and the strange unit are simply removed.

4) Chemical properties features

The CAS we got from ECOTOX are not a satisfactory representation of the chemicals since we miss information about their structures, so we need to move to a different form: the SMILES, (*"Simplified Molecular-Input Line-entry System"*), a line notation for describing the structure of chemical species. Once the SMILES specifications are obtained, meaningful information must be caught. This is possible thanks to the Python's library `rdkit`, which gets the following features:

1) Categorical: *Rings number*, *Double bonds number*, *Triple bonds number*, *Alone Atoms number*.
2) Non-categorical: *bonds number*, *Atoms number*, *Quantity of Mol*, *Morgan Density*, *Partition number (LogP)*.

C. Repeated Experiments analysis

Some experiments are repeated several times. Studying this phenomenon allows understanding the real experiment precision. For both classifications, we compute the accuracy considering as correct the result with the greatest number of occurrences among the repeated experiments (e.g. for binary classification, experiment repeated 5 times, having as results 3 "zeros" and 2 "ones", has an accuracy of 0.6). By computing the mean over the outcomes, an accuracy of **0.95** for binary classification and **0.87** for multiclass classification is found. In the Appendix (Fig. 4) two plots describing how the accuracy's mean change with the repetitions number are presented. From these plots we can notice how the accuracy falls with the number of repetitions for both cases. This behaviour underlines the limits of these experiments. Moreover, we can see how increasing the number of classes decrease the precision of the experiments. We decided to aggregate the repeated experiments taking the **median of the concentration values**, aware of the fact that our model will never be able to handle the experiments variance.

Once the features are chosen and the repeated experiments aggregated, we obtain the final dimension of our **base dataset: 23332** x **17 possible features**.

D. Features Transformation

Some of the non-categorical features need to be **transformed**: *bonds number*, *Atoms number*, *Quantity of Mol*. Indeed, from the plots of the features' distributions, it emerges that they need to be transformed because of their skewness and the high range of possible values. We decided to compute the $\log(1+x)$ and $MinMax(x)$ re-scaling transformations (applied in this order). An example of skewness we resolved this way is showed in Figure 2 (Plots for all the features in Appendix, Figure 3).

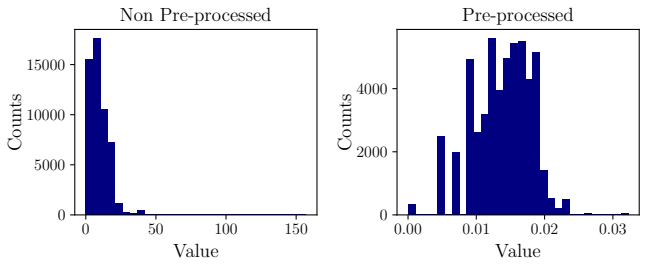

Fig. 2. Example of skewness for the atom number variable.

The categorical features do not need a direct transformation, but an encoding phase will be necessary to use them. As the type of encoding is strongly model-dependent, we leave this part to the analysis of the models, done in Section III and IV.

E. Score Attribution

The scope of this project is to predict the impact of each chemicals on a certain species. To do so, for the binary classification problem, we need to set a threshold on the concentration value. We set so the score to 1 if the concentration is above 1 mg/L and 0 otherwise. This value, widely accepted by the scientific community, also good divides the experiments. Since a regression problem is not well suited for this data (which has high variability in terms of repeated experiments, as said in Section II-C), we extended to multiclass classification. We define different thresholds in order to have five different scores classes, from the most powerful chemicals (score **5**), to the weakest ones (score **1**). The scores are summarized in TABLE I.

Concentration [mg/l]			
From	To	Binary score	Multiclass score
$-\infty$	10^{-1}	0	5
10^{-1}	10^{0}	0	4
10^{0}	10^{1}	1	3
10^{1}	10^{2}	1	2
10^{2}	∞	1	1

TABLE I
DEFINED CONCENTRATIONS SCORES

III. K-NEAREST-NEIGHBORS

A. Null models

We built the following two baseline models to have a benchmark: the first naive model **randomly predicts** the belonging class for each observation, the second one always predicts the **most frequent** class. The values of the accuracy that we can see in TABLE II reflect the fact that we chose the threshold values of our classes in order to make them equally distributed.

	Random	Majority
Binary	0.498	0.585
Multiclass	0.207	0.253

TABLE II
NULL MODELS RESULTS FOR K-NN

B. Model Selection

Before deciding to use k-NN, a preliminary phase of model selection was carried out. We tried to use other classification algorithms, such as Random Forests or Logistic Regression. Our features matrix is $X \in \mathbb{R}^{N \times D}$, where N is the number of datapoints and the D features can be either numerical or categorical. In a first approximation, the D features contained in the matrix are all the possible one, for a total of **17** features.

To treat the categorical features at this stage, a *"one-hot encoding"* on these features of X is performed and the three mentioned algorithms used. The hyper-parameters are not tuned at this stage, so the default ones from the `scikit-learn` library are used. The results are presented in TABLE III.

	k-NN	Random Forest	Logistic Regression
Binary	0.791	0.875	0.769
Multiclass	0.474	0.677	0.448

TABLE III
ACCURACY OF THE SELECTED SUPERVISED MODELS

The accuracy of all the models is quite good, with the Random Forest that clearly overcomes the others. However using a *one-hot* encoding increases the dimension to **899** features, which is quite big for the model to be efficient and interpretable. For these reasons, we decided to move to an **ordinal** encoding, where each category is just mapped to a number. This approach has a big drawback: using the classical loss functions treat these features as real numbers and the measured distance will be erroneous (e.g. if $species_1$ is mapped to 1 and $species_2$ to 500 a MAE loss will measure 499, while a good distance will be 1, simply meaning they are different). Due to our necessity of defining a different loss easily using the `scikit-learn` package and because the measured accuracy can be easily improved with a correct definition of the objective function, we chose the **k-NN** as our classification algorithm.

C. Model processing

In order to implement our model, we used the `scikit-learn` function `KNeighborsClassifier`. The `sklearn` implementation has the possibility to specify a symmetric matrix distance M such that $M_{i,j}$ represents the distance between observation i and observation j, instead of using the classical notion of distance implemented in the library. To take into account the numerical encoding we defined our distance function as

$$d(i,j) = \alpha d_1(i,j) + d_2(i,j)$$

where $d_1(i,j)$ is the *Hamming distance* between observation *i* and *j* (applied on encoded categorical features) while $d_2(i,j)$ is the *Euclidean distance* (applied on numerical features). So our distance matrix is $M = \alpha \cdot D_1 + D_2$ where D_1 is the distance matrix of the observations with the categorical features and D_2 is the distance matrix for the numerical ones.

In order to achieve the best performances from our model, two selections are performed: one **online features selection** phase and an **hyper-parameters selection** over α, *leaf size* and the neighbors number (K) by using a **3-fold** cross-validation (CV). As outcome of the first selection, all the **17 features** are selected to be used by the k-NN algorithm for both the classification and multi classification problem, with the aim to increase accuracy. The results of the CV search are reported in TABLE IV

	α	K	Leaf size
Binary	$1.61 * 10^{-3}$	1	60
Multiclass	$4.28 * 10^{-3}$	1	10

TABLE IV
HYPER-PARAMETERS SELECTION FOR K-NN

IV. MATRIX FACTORIZATION

Instead of using a *"classic"* technique as we did in the previous section, we now want to use a different perspective and use a **recommender system** to solve the problem. The basic idea is simple, the effect of the chemicals on the species is a **score** to each chemical/species pair: the lowest the concentration needed to achieve the desired effect, the highest the score. These scores can be considered as ratings, as it's usually done for recommender systems problems, hence the rational behind the choice of this algorithm.

Before getting inside the implementation, the mathematical foundations of the model are an important prerequisite. We use the Matrix Factorization (MF) method implemented in the `turicreate` **library**. It implements **global mean**, **biases**, **side features** and **regularization** as a plus with respect to the normal factorization algorithm. A score for chemicals i and species j is then computed as:

$$\widehat{r}_{ij} = \mu + w_i + w_j + \mathbf{a}^{\mathbf{T}}\mathbf{x}_i + \mathbf{b}^{\mathbf{T}}\mathbf{y}_j + \mathbf{u}_i^{\mathbf{T}}\mathbf{v}_j$$

The global mean parameter, μ, is the global mean among all the scores in the original matrix, the biases (or weight) terms

are the score mean for the given chemicals (w_i) or species (w_j). All of them are scalars [2]
. The side features are categorical and used to describe meaningful characteristics of either the chemicals, the species or the experiment itself. The algorithm embeds them by using two side features vectors, $\boldsymbol{x_i}$ and $\boldsymbol{y_j}$, containing the information for the chemical i and the species j, respectively. In the case of an experiment feature, this is added in both vectors. The two are fitted with a linear regression during the training phase, leading to two weight vectors $\mathbf{a}$ and $\mathbf{b}$[3].
The two vectors $\boldsymbol{u_i}$ and $\boldsymbol{v_j}$ are the rows of the $\mathbf{U}$ and $\mathbf{V}$ matrices, each of length number of **latent features**.
The training objective is defined as:

$$\min_{\mathbf{w},\mathbf{a},\mathbf{b},\mathbf{V},\mathbf{U}} \frac{1}{D} \sum_{(i,j,r_{ij})} (\widehat{r}_{ij} - r_{ij})^2 + \lambda_1(\|\mathbf{w}\|_2^2 + \|\mathbf{a}\|_2^2 + \|\mathbf{b}\|_2^2) + \lambda_2(\|\mathbf{U}\|_2^2 + \|\mathbf{V}\|_2^2)$$

Where D is the dataset dimension and r_{ij} is the true score. The model learning is regularized by using two different regularization parameters to avoid overfitting tendencies. The first (λ_1) is the *linear regularization* parameter and it's used to regularize the linear regression on the side features, the second (λ_2) is the *regularization* parameter and it is used for the two factorized matrices, to limit the number of latent features. The model is trained by using **Stochastic Gradient Descent**.

A. Baseline

The baseline used for this model is different from the previous. The factorization algorithm will produce as a score a real-valued number inside the range of the possible scores. For this reason, the most accurate metric for this model is not an accuracy but the **Root Mean Squared Error (RMSE)**. The baseline produced for this model is very simple: the prediction is drawn randomly from a normal distribution inside the chosen range. The accuracy of this model is anyway computed by rounding the predictions to the nearest integer, and this technique will be used also on the results from the factorization, in order to compare them with k-NN's accuracy. The results of the naive model are presented in TABLE V.

	RMSE	Acc
Binary	0.576	0.505
Multiclass	1.759	0.206

TABLE V
NULL MODELS RESULTS FOR MATRIX FACTORIZATION

B. Model processing

The scores are already defined Section II-E, so we just need to select which of our categorical features using as side features for experiments, chemicals and species, together with the correct hyper-parameters of the model. In order to do so, we performed an **online feature selection** phase, as we did for the k-NN. The results are slightly different for the two different problems:

1) *Experiments*: *exposure type*, *concentration type* and *observation duration* are chosen for both the classifications as they correctly represent the experiment settings with only a few possible categories.
2) *Chemicals*: *Triple bonds number* is the only feature selected.
3) *Species*: *class*, *tax order* and *genus* are selected for the classification. For the multiclass problem, also *family* is chosen.

No one of these features needs encoding in a preliminary phase, as this is already done by `turicreate` implementation.
The hyper-parameters selection is performed by using a **3-fold** cross-validation, with decreasing RMSE as the target to select the best parameters. The best choices for both models are reported in TABLE VI.

	Latent Factors	λ_1	λ_2
Binary	9	$3.73 * 10^{-11}$	$1.18 * 10^{-5}$
Multiclass	9	1.93^{-6}	$4.39 * 10^{-4}$

TABLE VI
HYPER-PARAMETERS SELECTION FOR MATRIX FACTORIZATION

V. RESULTS & DISCUSSION

		Accuracy	RMSE
k-NN	Binary	0.903	0.312
	Multiclass	0.740	0.687
MF	Binary	0.906	0.272
	Multiclass	0.662	0.637

TABLE VII
BEST RESULTS FOR OUR ML ALGORITHMS

The final results, showed in TABLE VII, confirm the expectation we had. The algorithms **overcome** their respective baselines both for the binary and the multiclass classification. For the **binary classification** case, the accuracy of the two models are similar, symptoms that the problem is correctly defined and interpretable. The same thing happens for the RMSE, but the MF's one is lower, as expected, due to the prediction in a real-valued range instead of the integer one.
The **multiclassification case** doesn't have the same precision as the binary one. An important comparison between the two algorithms can anyway be done in terms of accuracy and RMSE. As expected, the RMSE is better for the MF case, but this result is not reflected in the accuracy, which is higher for the k-NN. This means that both algorithms do very well in the **optimization toward their respective objective measures**, but there's a trade-off with the other one. *A note on the MF results*: as the algorithm is highly unpredictable, with slightly different results even with the same parameters, we run it more times and we took the mean over the runs.

When interpreting these results, the **real-world accuracy** must be taken into account too. As we stated in Section II-C, even in the repeated experiments there is an uncertainty that must be considered and this is reflected in the data we used to train our model. Better data could improve our results.

In conclusion, we can state that the results achieved in this project, taking into account the short time frame and the initial data quality, meet our expectations both in terms of correctness and interpretability. Of course, there are a lot of aspects that could be improved and extended. However, what we state in this project can be a fundamental starting point for any next research in this direction, like using more sophisticated models or gather additional features from new data sources.

REFERENCES

[1] ECOTOX. Homepage. [Online]. Available: https://cfpub.epa.gov/ecotox/help.cfm

[2] R. B. Yehuda Koren and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Computer Society*, pp. 42–49, 2009.

[3] A. A. Ian Porteous and M. Welling, "Bayesian matrix factorization with side information and dirichlet process mixtures," *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10)*.

APPENDIX

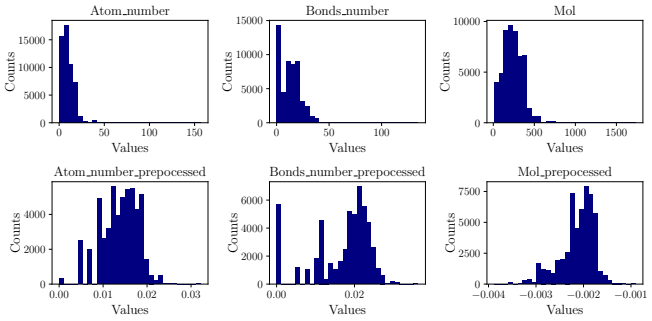

Fig. 3. Skewness of the numerical features.

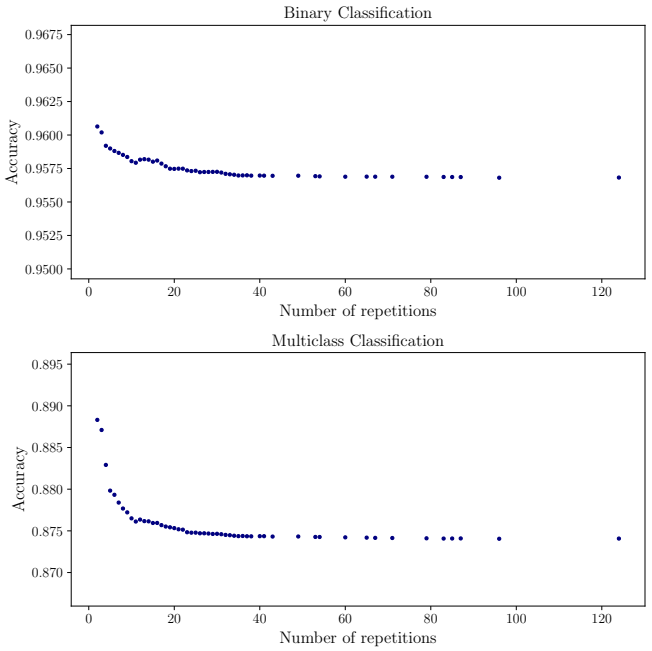

Fig. 4. Experiments accuracy trend with number of experiments.