Machine Learning - Home Assignment 2

2022/2023

Group 6 – Alexandre Sobreira (59451) André Dias (59452) Nihan Ahat (No assigned number) Tiago Rodrigues (49593)

Hours contributed – Alexandre: 15h; André: 15h; Nihan: 15h; Tiago: 15h

Persistent, bioaccumulative, and toxic (PBT) chemicals are a group of chemicals that are harmful to the environment and human health. These chemicals are characterised by their ability to remain in the environment for long periods, accumulate in living organisms, and cause adverse effects at low levels of exposure. In addition, PBT chemicals are often resistant to degradation, making them difficult to remove from the environment once released. Biodegradation is the process by which microorganisms break down chemical substances, such as pollutants and waste, into simpler, more environmentally friendly compounds. This process is vital to reduce the amount of waste and pollution in the environment. PBT chemicals are not typically biodegradable, as they are resistant to degradation. However, some limited chemicals have been tested for their degradability.

In this study, different classification methods will discriminate between biodegradable (RB) and non-biodegradable (NRB) chemicals. The data for this study was obtained from Mansouri's publicly-available QSAR biodegradation dataset. This dataset contains quantitative chemical analysis for 1055 chemical compounds and 41 columns (attributes). The variables in the data are either categorical ('B01'; 'B03'; 'B04'; 'Biodegradable') or continuous/discrete.

The objective of this study is to use QSAR-Data to build classification models to decide if chemicals are biodegradable or non-biodegradable. By looking at molecular properties, it is possible to estimate the behaviour of these chemicals (in terms of their ability to degrade).

It is considered that a dataset is unbalanced if the target variable has more observations in one class than in another. In the dataset provided, the dependent variable ('Biodegradable') is unbalanced because there are a lot more instances of biodegradable than non-biodegradable (approximately 80-20% split). Given that the dependent variable is unbalanced, the F1 metric is considered a better indication of the model's quality since it combines precision and recall and will be regarded as the primary metric to evaluate the quality of the models tested in this work.

Before any modelling could be done, the dataset needed to be pre-processed, i.e., search and deal with missing values and duplicates, and, if required, apply scaling and variable codification.

After some preliminary analysis of the dataset structure and statistics, it was observed that the whole dataset contained 7056 missing cells and two duplicate rows. The features with missing values were the following: 'F01', 'C', 'nCp', 'HyWi_B', 'F03_CO', 'Me', 'nCIR', 'SpMax_A', 'SdO', 'nCrt', 'SpMax_B', 'Psi_i_A', 'nX'.

Since a high quantity of missing cells was observed, row dropping was not considered since it would dramatically reduce the dataset's size and decrease the information available. With this in mind, two alternative approaches were applied.

The first approach consisted in maintaining all features and inputting the missing values. For the imputation, two criteria were considered: (1) The distribution of the variable: for asymmetrical or skewed distributions, the best measure of localisation was considered to be the median, while for symmetrical distributions, the mean was regarded as the most appropriate; (2) The type of variable: the median would be more appropriate for discrete variables, while for continuous variables the mean was considered the more appropriate measure for imputation, so the variables would not lose their specific properties.

The second approach was a kind of feature selection, where some features were removed according to specific characteristics. The ones' not removed were imputed according to the criteria defined in the previous approach. The removal of the features took into account two criteria: (1) Distribution with a high variance; (2) High percentage of missing values (> 10%). When these were observed, imputation was not performed since a high percentage of missing values or a distribution with high variance may lead to a biased imputation.

Before proceeding, the features with missing values were plotted (box plots) to visualise their distribution (Figure 1). These features' unique values were also inspected to check if they were discrete or continuous.

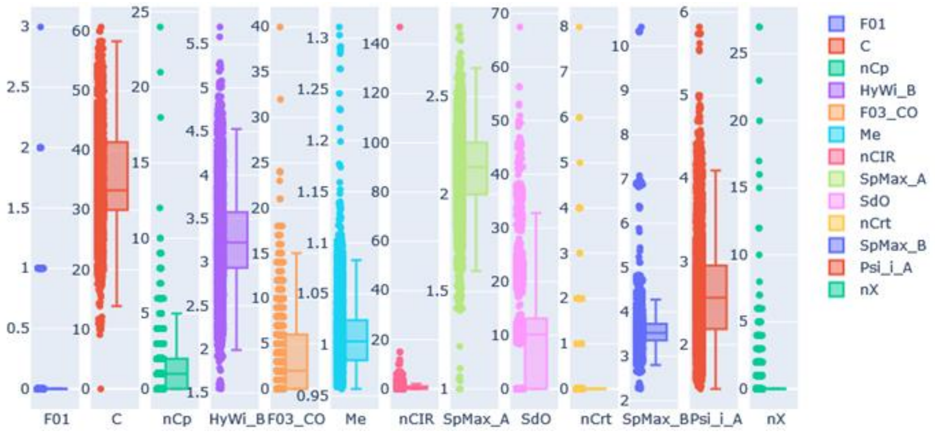


Figure 1 *– Box plot for each feature that contained missing values.*

Regarding the first approach, the following features were imputed with the median: 'F01' (discrete/skewed); 'nCp' (discrete/skewed); 'F03_CO' (discrete/skewed); 'nCIR' (discrete/skewed); 'SdO' (skewed); 'nCrt' (discrete/skewed); 'nX' (discrete/skewed); 'SpMax_B' (skewed); 'Psi_i_A' (skewed); 'Me' (skewed). The rest of the variables were imputed with the mean: 'C' (continuous/close to symmetrical); 'HyWi-B' (continuous/close to symmetrical); 'SpMax_A' (continuous/close to symmetrical). The dataset derived from this approach was denominated as 'Data'. Before proceeding, the duplicated rows were dropped (two rows)

For the second approach, the distribution and percentage of missing values were examined. The following features were removed: 'C', with 16.8% missing values and high variance; 'HyWi_B', with 10.5% missing values and high variance; 'SpMax_A', with 14.8% missing values; 'SdO', with high variance; 'SpMax_B', with 29.8% missing values and high variance; 'Psi_i_A', with 10.1% missing values and high variance. The remaining features with missing values were imputed with the median, given they all were discrete or skewed ('F01'; 'nCp'; 'F03_CO'; 'nCIR'; 'nCrt'; 'nX'; 'Me'). The dataset derived from this approach was denominated as 'Less_data'.

For proper model validation, both datasets were divided into two separate dataframes: y ('Data') and y_less ('Less_data'), both containing 'Biodegradable' and X ('Data') and X_less ('Less_data'), both with the remaining features from each dataset. Next, both datasets were split into a training set (75%) and a test set (25%).

Categorical variables usually require codification so that classification models can be adequately applied. One possible method is One Hot Encoding which transforms the variables into binary information. However, the categorical independent variables in both datasets ('B01'; 'B03'; 'B04') did not require any codification, given they were already in binary format. For the dependent variable ('Biodegradable'), 'RB' was replaced with the value 1 and 'NRB' with the value 0.

Furthermore, given the different metrics' variances and units in the dataset, both datasets were scaled with the intention of an equal comparison between models. A 'StandardScaler' was first fitted to X_train and X_train_less and applied to both X_train / X_train_less and X_test / X_test_less.

With all the data pre-processed, the next step was to compare both datasets and observe both approaches' impact on F1-scores. Random Forest (RF), Extreme Gradient Boost (XGB), Logistic Regression (LR) and K-Nearest Neighbours (KNN) models were chosen for this. The default hyperparameters were maintained in all models, and simple cross-validation was applied. Table 1 compares the F1-scores obtained by the four models when applied to both datasets.

Table 1 shows that, generally, only a slight decrease in F1-scores from the first dataset ('Data') to the second one ('Less_data') occurs, revealing that the features dropped in the second approach were likely of reduced importance in classifying the dependent variable ('Biodegradable'). This may be considered a fair trade-off, given that six features with high missing-value percentages were dropped, making the models applied to 'Less_data' possibly less biased, more parsimonious, and easier to explain. Thus, this became the primary dataset for the forthcoming procedures.

Table 1 *– F1-scores obtained through simple cross validation for 'Data' and 'Less_data' according to four models.*

Models	Data	Less_data
RF	0.9822	0.9837
XGB	0.9838	0.9806
LR	0.9719	0.9704
KNN	0.9754	0.9718

After the initial feature selection, the 'Less_data' dataset had 36 independent variables. However, despite this initial selection step, there can still be many variables that do not contribute significantly towards the models and increase their complexity unnecessarily. As such, it was deemed essential to find the minimal set of relevant features that provide a balance between the best possible prediction and the least number of features by applying a proper feature selection.

Since there are several approaches for feature selection, some methods were applied to the 'Less_data' dataset. These approaches were the analysis of correlation (or Variance based Feature Selection), Stepwise (Forward and Backward methods), and ensemble machine learning methods (using Random Forests). From the results obtained for each approach, a comparison was performed and if a variable appeared in three approaches, it would be selected to create a new dataset with only these selected variables called 'Feature_selection_data'.

The basis of the correlation feature selection approach is that all the variables that correlate with the dependent variable ('Biodegradable') below a chosen threshold would be removed. For this, the *numpy.corrcoef* function was used to get the Pearson product-moment correlation coefficients. The correlation threshold chosen for the approach was 0.3, which selected 12 features.

For Stepwise methods, the main idea is to iteratively change the base models by selecting the best feature to consider/take out next. Within Stepwise, Forward Selection and Backward Selection exist. Forward Selection tries each feature individually and then adds to the model the one that gave it the best performance. With that new model, a new feature is selected from the available ones that provide a better performance, and this process repeats itself until there are no more features left. For Backward Selection, instead of beginning with the model with a single feature, a model with all the features is firstly considered. Sequential removal of features is then performed on the model and its performance is verified. Both approaches operate on an identical basis, which is to identify the number of features after which performance is no longer gained/lost.

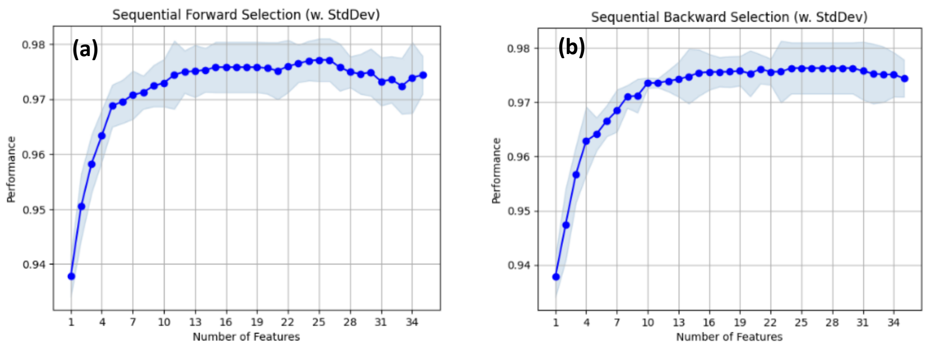
Since this approach can use many models as basis to perform its calculations, it was decided to use AdaBoost for both Forward and Backward Selection. The performance of each model iteration was then plotted for both Forward and Backward Selection (Figure 2a and b respectively). For Forward Selection, it is possible to observe that with 11 features, the model's performance reaches a plateau, which means that there aren't any relevant performance improvements with the addition of more features, making it the ideal amount of features to select. For the results of Backward Selection, the plateau is reached at around 11 as well.

Lastly, Random Forests is one of the standard ensemble models for machine learning methods. This approach lets us get some insight from the data, identify which features are more important and contribute the most to the overall model quality. This approach selected a total of 15 features because of their higher importance.

All the methods results' were compared (Figure 2c), and out of the 36 variables, eight were chosen ('SpMax_L', 'nHM', 'F04', 'NssssC', 'nCb', 'F03', 'F02_CN', 'SM6_B').

A new dataset containing the selected features was created from this feature selection phase ('Feature_selection_data'). Again, for proper model validation, X_feature_selection and y_feature_selection were split into a training set (X_train_feature_selection and y_train_feature_selection) (75%) and a testing set (X_test_feature_selection and y_test_feature_selection) (25 %). Furthermore, the codification of the dependent variable was performed, and the scaling was applied similarly to what had been done to 'Data' and 'Less_data'.

After feature selection, several classification models were tested for each of the datasets ('Data'; 'Less_data'; 'Feature_selection_data'). The classification models were first tested using the parameters' default values, and afterwards, GridSearchCV was applied to the models using different sets of parameters. The parameters used in each model and their ranges are stated in **Table 2**.



(c)

Correlation (> 0.3)	index	**0**	**2**	**4**	**6**	**9**	**19**	**22**	**28**	**29**	**31**	**32**	**34**	-	-	-
	feature name	SpMax_L	nHM	F04	nCb	F03	SpPosA_B	B03	C_026	F02_CN	nN	SM6_B	nX	-	-	-
Forward	index	**2**	**5**	**6**	**7**	**8**	**9**	**17**	**27**	**29**	**30**	**31**	-	-	-	-
	feature name	nHM	NssssC	nCb	nCp	nO	F03	nArNO2	nCrt	F02_CN	nHDon	nN	-	-	-	-
Backward	index	**0**	**1**	**2**	**4**	**5**	**6**	**10**	**15**	**26**	**27**	**32**	-	-	-	-
	feature name	SpMax_L	J_Dz(e)	nHM	F04	NssssC	nCb	SdssC	Mi	TI2_L	nCrt	SM6_B	-	-	-	-
Random Forest	index	**0**	**1**	**2**	**4**	**5**	**6**	**9**	**11**	**12**	**15**	**19**	**20**	**29**	**32**	**34**
	feature name	SpMax_L	J_Dz(e)	nHM	F04	NssssC	nCb	F03	LOC	SM6_L	Mi	SpPosA_B	nCIR	F02_CN	SM6_B	nX

Figure 2 *– Stepwise approach for Forward (a) and Backward (b) Selection. Features selected from each methodology (c). Each selected feature is coloured to identify in which models they were selected.*

Table 3 *– Parameters used for the GridSearchCV of each classification model used. DT – Decision Tree; LR – Logistic Regression; NB – Naïve Bayes; KNN – K-Nearest Neighbors; LSVM – Linear Support Vector Machine; SVM – Support Vector Machine; BAG – Bagging; RF – Random Forest; ADA – AdaBoost; GB – Gradient Boost; XGB – Extreme Gradient Boost.*

Model	GridSearchCV Parameters
DT	criterion: ['gini', 'entropy'], max_depth: np.arrange(1, 26, 2), max_leaf_nodes: np.arrange(5, 71, 5), splitter: ['best', 'random'], min_samples_leaf: np.arrange(1, 26, 2)
LR	penalty: ['l1', 'l2', 'elasticnet'], C: [0.001, 0.01, 0.1, 1, 10, 100, 1000]
NB	var_smoothing: np.logspace(0, -9, num = 100)
KNN	n_neighbours: np.arrange(3, 14, 2), weights: ['uniform', 'distance'], metric: ['euclideand', 'manhattan', 'minkowski']
LSVM	penalty: ['l1', 'l2'], C: [0.01, 0.1, 1, 10, 100], max_iter: [1, 10, 100, 1000, 10000]
SVM	gamma: ['scale', 'auto', 0.01, 0.1, 1, 10, 100], C: [0.01, 0.1, 1, 10, 100], kernel: ['linear', 'rbf', 'sigmoid']
BAG	base_estimator: [DecisionTreeClassifier(), GaussianNB(), LinearSVC(), SVC()], n_estimators: [1, 5, 10, 25, 50, 100], max_samples: [0.5, 0.75, 0.9, 1], max_features: [0.5, 0.75, 0.9, 1]
RF	n_estimators: [10, 50, 100, 500], criterion: ['gini', 'entropy', 'log_loss'], max_depth: [5, 10, 20, None], min_samples_split: [2, 5, 10], min_samples_leaf: [1, 5, 10], max_leaf_nodes: [50, 100, 200, None]
ADA	n_estimators: [10, 25, 50, 100, 150, 200], learning_rate: [0.01, 0.1, 1, 1.5, 2, 3]
GB	n_estimators: [10, 25, 50, 100, 150, 200], learning_rate: [0.01, 0.1, 1, 1.5, 2, 3], max_depth: [1, 3, 5, 10, 25, 50]
XBG	n_estimators: [50, 100, 200], learning_rate: [0.1, 0.5, 0.75, 1], min_child_weight: [1, 5, 10], gamma: [0, 0.5, 1, 1.5, 2, 5], max_depth: [3, 6, 8, 10]

From the GridSearchCVs, the best parameters for each model were identified (**Table 3**):

Table 2 *– Best parameters obtained from the GridSearchCV for each of the tested data structures.*

Model	Best Parameters ("Data" | "Less_data" | "Feature_selection_data")
DT	criterion: 'gini'|'gini'|'gini', max_depth: 15|13|13, max_leaf_nodes: 45|45|45, splitter: 'best'|'best|'best', min_samples_leaf: 1|1|1
LR	penalty: 'l2'|'l2'|'l2', C: 10|1|1
NV	var_smoothing: 0.0053|0.0123|0.1874
KNN	n_neighbours: 3|3|5, weights: 'distance'|'distance'|' distance'', metric: 'minkowski'|' minkowski'|'minkowski'
LSVM	penalty: 'l2'|'l2'|'l2', C: 10|1|0.1, max_iter: 1000|1000|100
SVM	gamma: 'scale'|'scale'|'scale', C: 10|10|10, kernel: 'rbf'|'rbf'|'rbf'
BAG	base_estimator: 'DecisionTreeClassifier()'|DecisionTreeClassifier()|DecisionTreeClassifier(), n_estimators: 50|50|50, max_samples: 0.9|0.75|0.75, max_features: 0.9|0.9|0.75
RF	n_estimators: 100|10|50, criterion: 'gini'|'gini'|'entropy', max_depth: None|20|20, min_samples_split: 10|5|5, min_samples_leaf: 1|1|1, max_leaf_nodes: None| None| None
ADA	n_estimators: 200|200|200, learning_rate: 1.5|1.5|0.1
GB	n_estimators: 200|150|50, learning_rate: 0.5|0.1|0.1, max_depth: 5|10|5
XGB	n_estimators: 50|50|200, learning_rate: 0.75|0.5|0.1, min_child_weight: 1|1|1, gamma: 0.5|0|0.5, max_depth: 8|10|8

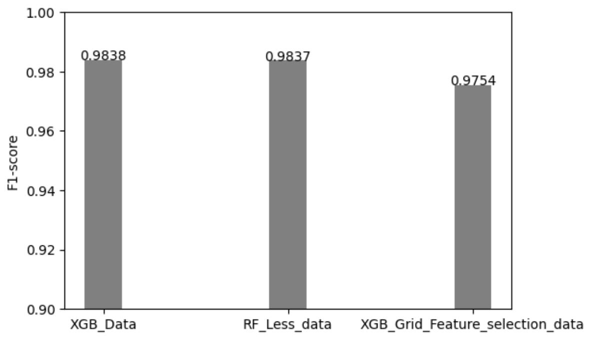
The F1-score obtained for each model with and without tuning can be seen in **Table 4**.

Table 4 *– F1-scores obtained for the different data structures using several classification models before and after tunning*

Model	**F1-score according to the Data Structure**					
	Data		*Less_data*		*Feature_selection_data*	
	Default Model	GSCV model	Default Model	GSCV model	Default Model	GSCV model
DT	0,9744	0,9775	0,9674	0,9759	0,9564	0,9712
LR	0,9719	0,9719	0,9704	0,9704	0,9634	0,9634
NB	0,9616	0,9635	0,9611	0,9630	0,9492	0,9625
KNN	0,9754	0,9790	0,9718	0,9765	0,9693	0,9723
LSVM	0,9709	0,9739	0,9709	0,9709	0,9631	0,9636
SVM	0,9760	0,9796	0,9791	0,9781	0,9705	0,9710
BAG	0,9765	0,9811	0,9764	0,9800	0,9690	0,9723
RF	0,9822	0,9806	0,9837	0,9832	0,9727	0,9728
ADA	0,9678	0,9723	0,9704	0,9718	0,9622	0,9635
GB	0,9770	0,9816	0,9770	0,9827	0,9714	0,9734
XGB	0,9838	0,9806	0,9806	0,9806	0,9728	0,9754

The presented scores make it possible to assess that the best overall model is Extreme Gradient Boost using the 'Data' dataset. Despite this, an important conclusion that can also be made is that only a small loss in the F1-score is observed when going from the 'Data' dataset, with 41 variables, to the 'Feature_selection_data' dataset, with 8. Frequently, a small trade in the model score in favour of model simplification is desired, as they are easier to compute and explain in a real-world scenario. It's also possible to visualise that some models lose performance after tuning, which would not be expected at first glance. These results can arise from the fact that GridSearchCV uses a K-Fold Cross Validation, which minimises data splitting bias compared to the employed Simple Cross-Validation. As such, it could be that the data split applied to the data to make the Simple Cross Validation may result in slightly biased training data, leading to the results seen.

Further analysis was conducted on the best model for each of the datasets: XGB ('Data'), RF ('Less_data') and XGB with GridSearchCV ('Feature_selection_data'). The confusion matrixes and F1-scores of these models can be seen in **Figure 3**.



XGB_Data

	0	1
0	171	23
1	8	939

RF_Less_data

	0	1
0	172	22
1	10	937

XGB_Grid_Feature_selection_data

	0	1
0	161	33
1	14	933

Figure 3 *– F1-scores and confusion matrixes for the three datasets obtained using their respective best model.*

From the confusion matrixes, it is possible to visualise the number of misclassifications and assess their differences according to each dataset. The 'Data' dataset had 31 misclassifications, accounting for 2.72% of all the classifications made. For 'Less_data', 32 misclassifications were made, resulting in a 2.80% misclassification rate. In the 'Feature_selection_data', 47 misclassifications were made, which translates to a misclassification rate of 4.12%. It's possible to see that 'Data' and 'Less_data' are almost identical, meaning that the variables discarded from 'Data' to 'Less_Data' were not relevant. Furthermore, from either of the two first datasets to the 'Feature_selection_data', a slight increase in the false negatives and false positives was seen, but this only amounted to 1.3 – 1.4% difference in misclassifications. This is also supported by the almost identical F1-scores of 'Data' and 'Less_data' and slight inferior score of the 'Feature_selection_data'. As previously mentioned, this may be a good trade-off for the potential reduction in model complexity and increased model explainability that accompanies the remarkable decrease in the variables used for the model (from 41/36 to a total of 8 variables).