Python for Data Analysis

Nicolas Vanderstigel & Arnaud Schwartz

Our subject : QSAR biodegradation Data Set

Our dataset contains values for 41 attributes (molecular descriptors) used to classify 1055 chemicals into 2 classes : ready and not ready for biodegradation.

Although our subject is interesting, many of the variables are highly complex and can’t be understood without a great expertise in chemistry. This complexity has led us to try to understand only some of the variables when they created interest for us in the analysis, rather than all the variables at the beginning.

Loading and Cleaning the Dataset

We had the luck to get a Dataset containing:
- No NaN values.
- Only numeric values (except our target, called “experimental class”, which was RB/NRB, and that we replaced with 1/0).

The only modification that me made was to scale and normalize our dataset, which is better for the upcoming algorithms.

Exept that, we have chosen to keep all the explaining variables untouched, to keep as much information as possible.

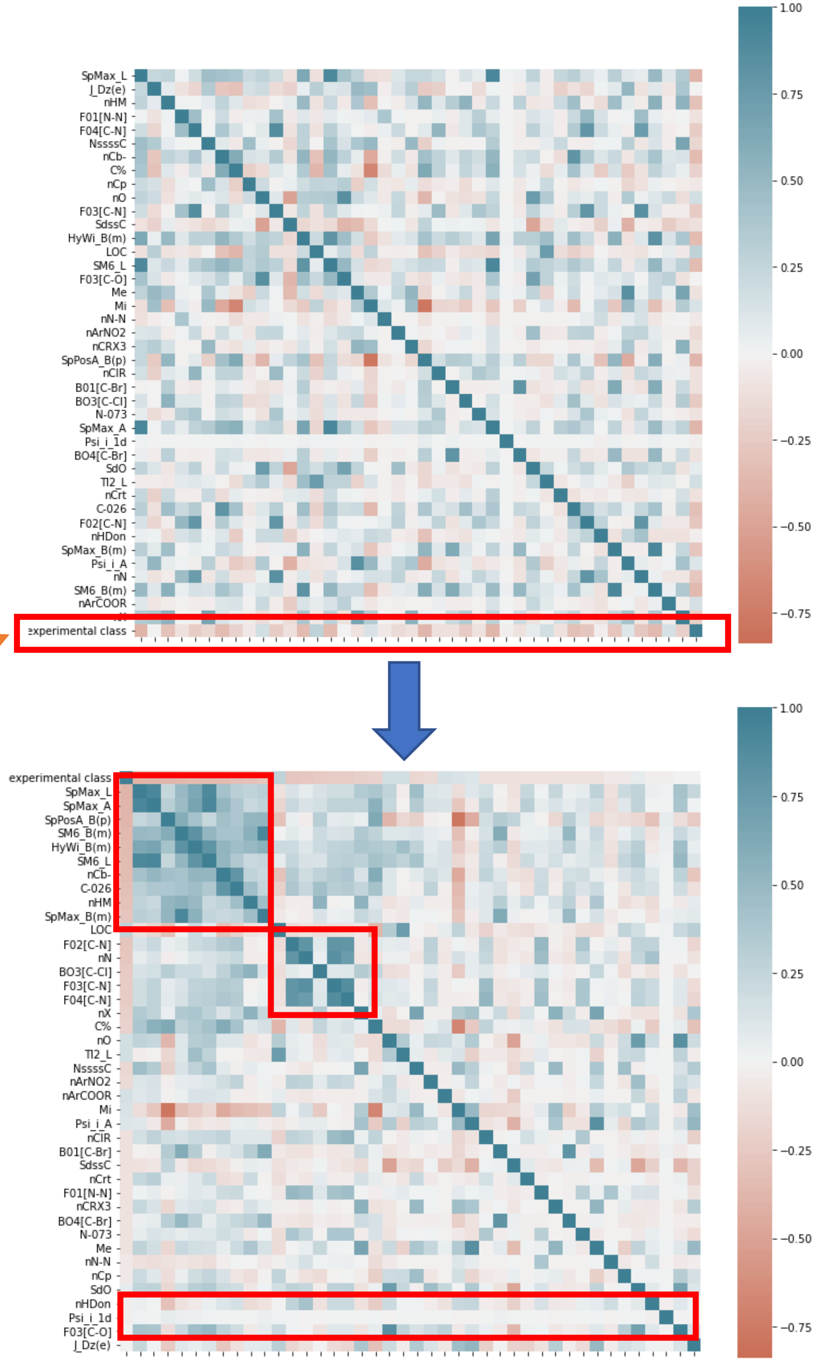
Understanding the Dataset

Our first instinct was to draw the correlation matrix.

Because of the orange color of the « experimental class » line, we clearly saw that the explaining variables where mainly negatively correlated with our target.

However, the matrix was difficultly understandable, so we ordered the variables in ascending order of the correlation.

This representation helped us to understand a lot more about the dataset, you can find every analysis /visualization about that in our Jupyter notebook.

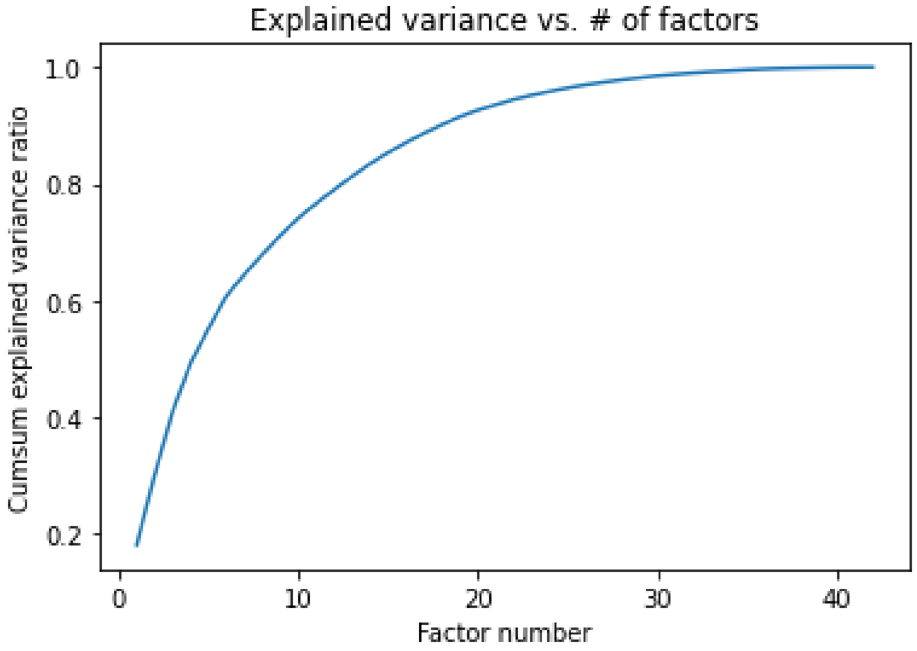


Principal Components Analysis

Knowing that we had more than 40 variables, we wanted to build an ACP to create a simpler dataset for visualisation (by reducing the number of dimensions).

However, the PCA wasn't very effective :

We couldn’t reach 95% of the explained variance with less than 30 dimensions. We concluded that the PCA was useless for the next steps, but it still gave us the information that pretty much every variable is important in our dataset and leads to information.

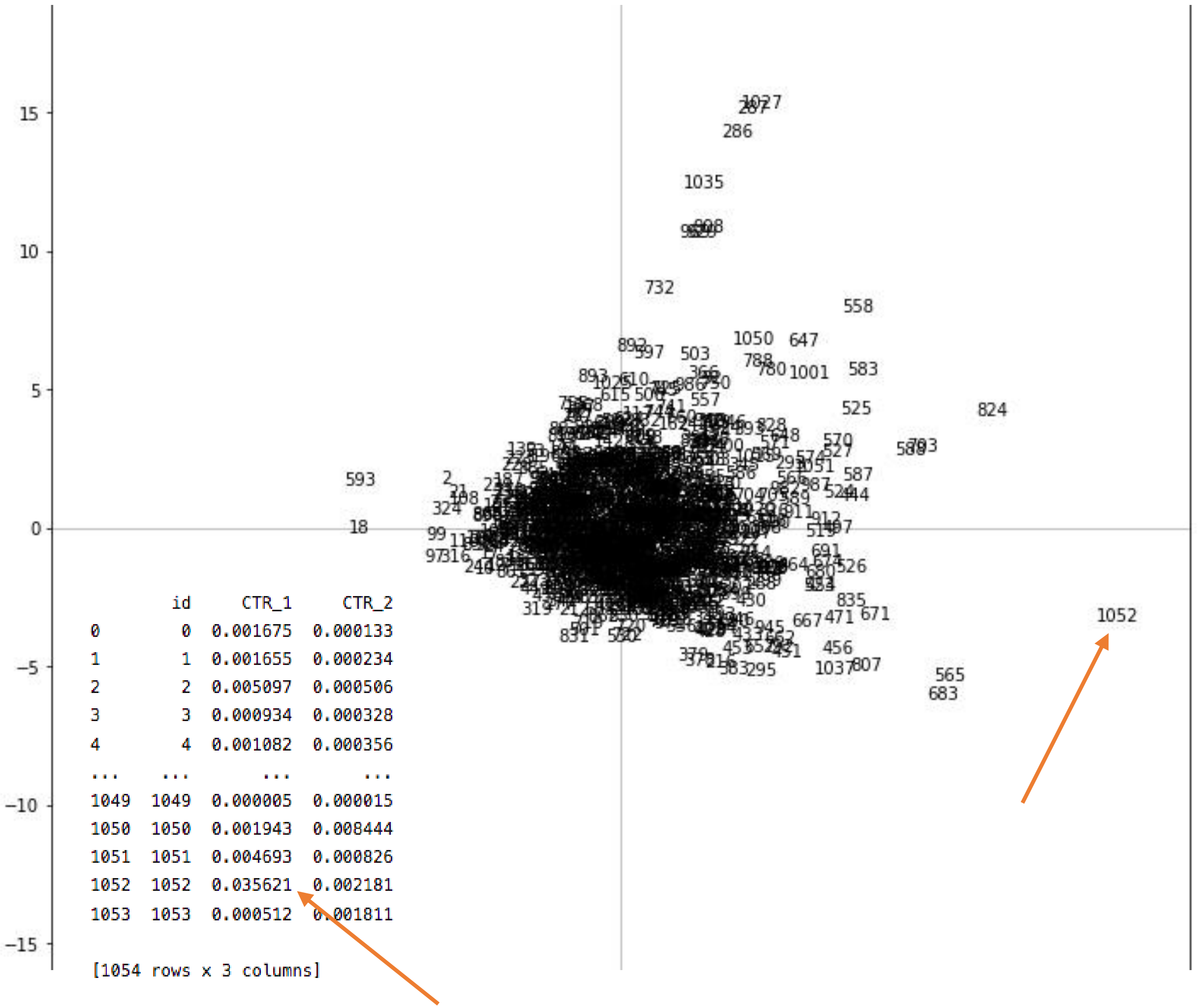


Working on features and observations

We wanted to show how every observations impact our PCA representation. For example, let's have look to the 1052nd observation.

It is far away from the others, and on the array of contributions we can see that it contributes around 3.5% to the first axe which is 10 times more compare to others.

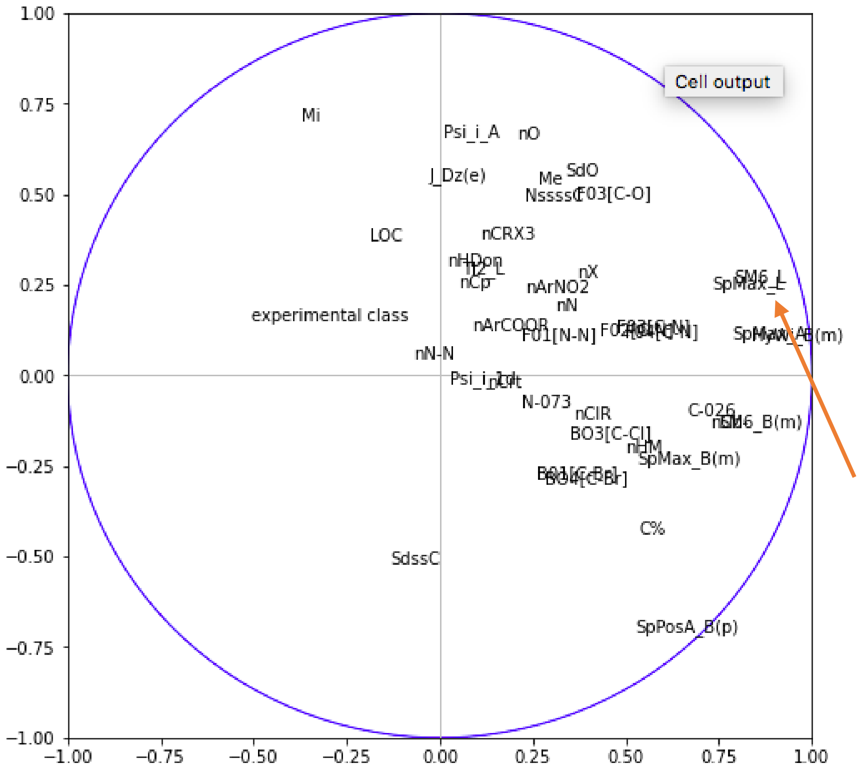
Finally, this is the only 2 dimensions representation that we can do.



```
        id     CTR_1     CTR_2
0        0  0.001675  0.000133
1        1  0.001655  0.000234
2        2  0.005097  0.000506
3        3  0.000934  0.000328
4        4  0.001082  0.000356
...    ...       ...       ...
1049  1049  0.000005  0.000015
1050  1050  0.001943  0.008444
1051  1051  0.004693  0.000826
1052  1052  0.035621  0.002181
1053  1053  0.000512  0.001811

[1054 rows x 3 columns]
```

Working on features and observations



Here is the correlation circle, more a feature is close to the circle, mean that our feature can be used in a bidimensional analysis. For example, SpMax_L could be used because it is very close to the circle.

It is important to notice that this graph can be linked with our correlation matrix because all the features that explain the most our target feature also influence the most this graph.

Finally, this analysis can also help us to understand which features are more important than others such as said before.

Machine Learning : The beginning

In our subject, the group of researchers used 3 different algorithms :

K Nearest Neighbours (KNN), Partial Least Squares Discriminant Analysis (PLS) and Support Vector Machines (SVM).

We choose to use those 3 models, plus a fourth : SGDClassifier (SGDC). SGDC is recommended for very large number of individuals, but we wanted to try it to compare its performance with the others.

To feed the algorithms, we built a train and a test set, with the same proportionality of Ready Biodegradable and Not Ready Biodegradable in both.

Machine Learning : The Methodology

For each model, we used the same steps :

Grid Search : Finding the best parameters for the model

Fitting the model : We used the SKLearn library all along the project

Results analysis : Creation of the confusion matrix, f1 score, accuracy…

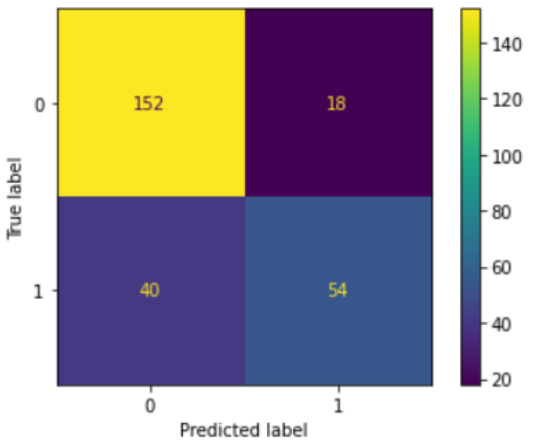
Storing : Storing the results in variables with explicit names for further analysis later : Probabilities to get a 0/1 as “y_proba_XXX”, predictions as “y_pred_XXX”,Confusion matrix as “cm_XXX”, f1 score as “f1_XXX”

(Knowing that “XXX” is the name of the model)

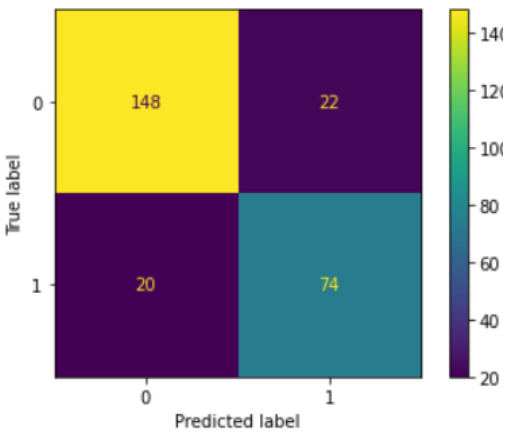
Machine Learning : Results

Here you can see the confusion matrix of each model :

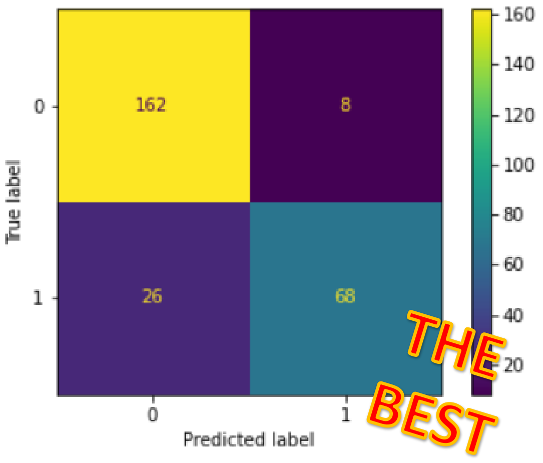
PLS



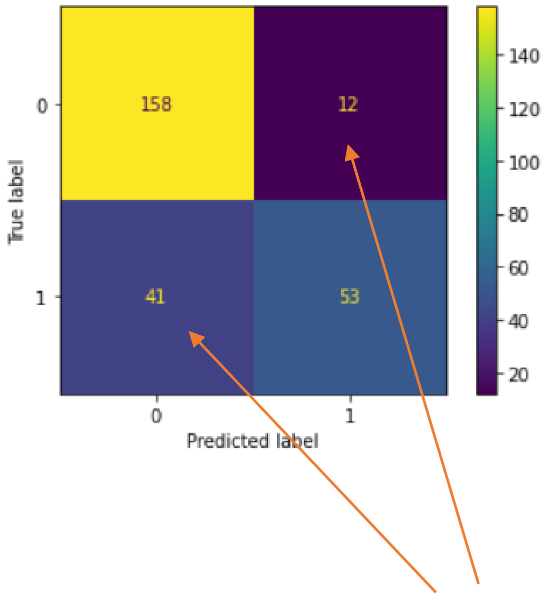
KNN



SVC



SGDC

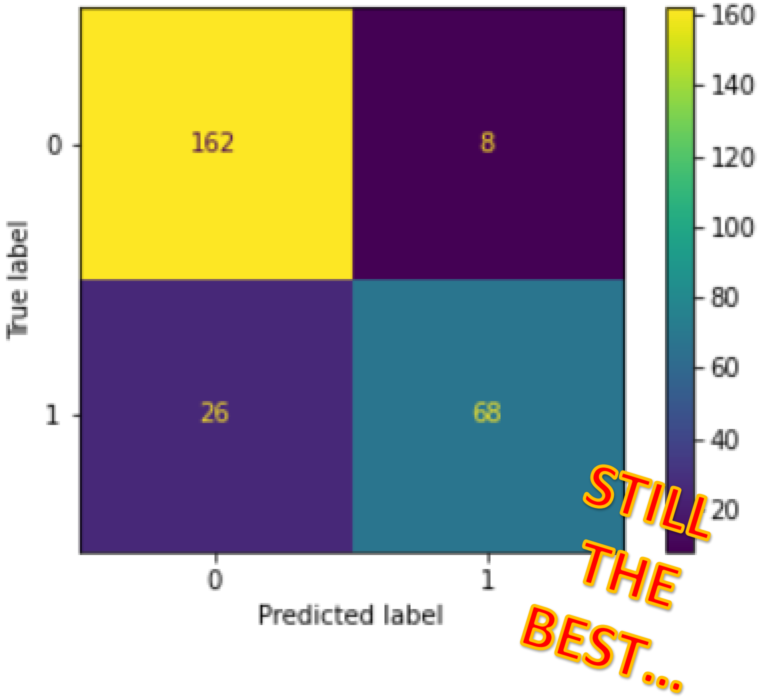


The fewer the false values are, the best the model is. We can see their count here.

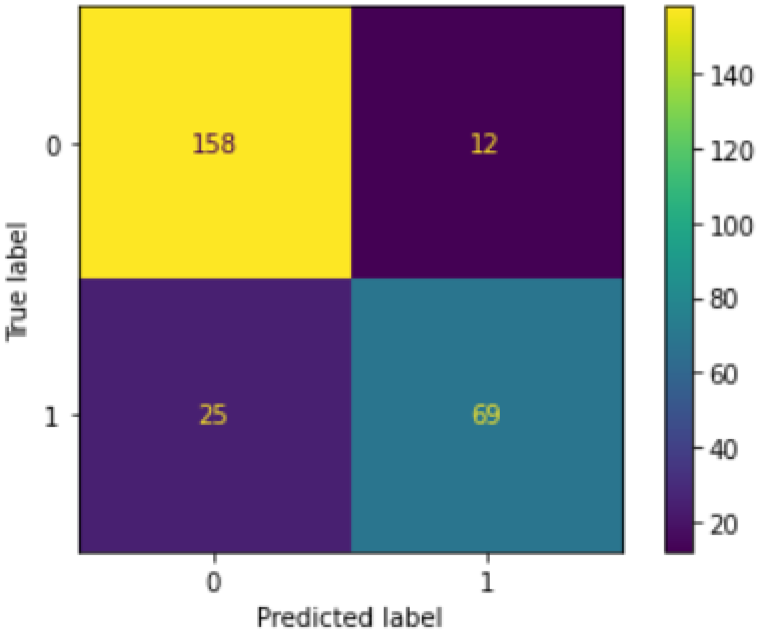
Machine Learning : Exploration

We tried to merge each of our models to get a more precise one. For that, we did the mean of the probability to gat a 1 for each individual of each method. In the end, we got this performance :

SVC



MERGED

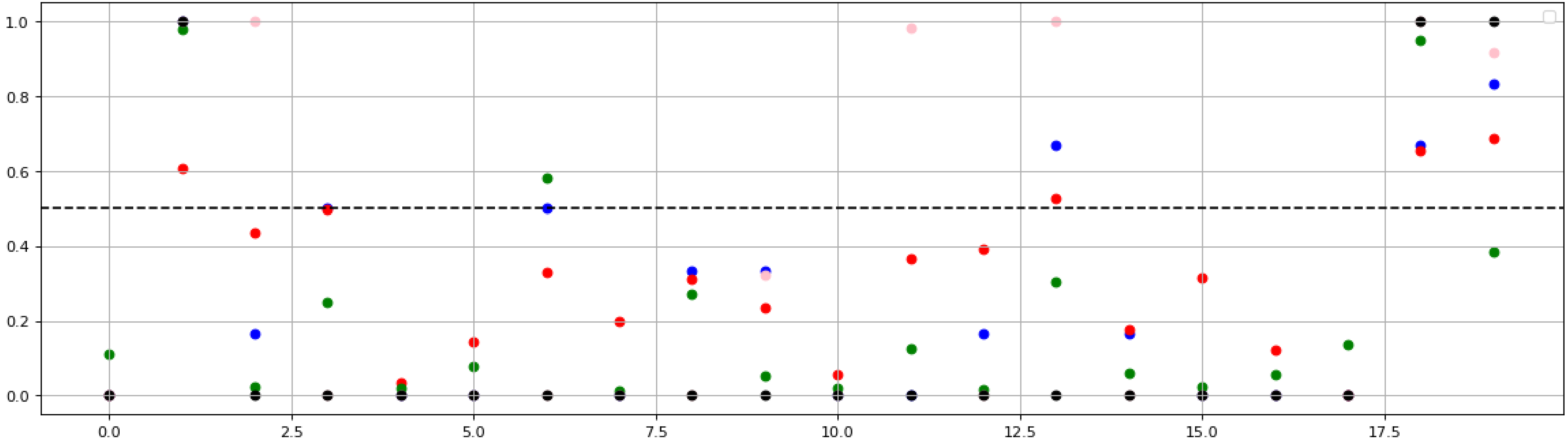


Sadly, the merged method isn’t as effective as SVC…

Machine Learning : Conclusion

Here you can find the probability for each algorithm, for the 20 first individuals.

PLS ● KNN ● SVC ● SGDC ● GLOBAL ●



1.0
0.8
0.6
0.4
0.2
0.0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5