Challenge Medichal – Groupe Parasite

Team members:
_Fellah Yanisn, yanis.fellah@u-psud.fr
_Kolczynski Samuel, samuel.kolczynski@u-psud.fr
_Djebarra Kahina, kahina.djebarra@u-psud.fr
_Bouamara Nabila, nabila.bouamar@u-psud.fr
_Rabiai Samir, samir.rabiai@u-psud.fr
_Trinh Duc-Bao, duc-bao.trinh@u-psud.fr

Challenge URL: **https://codalab.lri.fr/competitions/625**
Github repo of the project: **https://github.com/yanis-fellah/Parasite**

Introduction

The Medichal project is a datascience project where we are given images of cells and have to determined whether the cells are parasitized by malaria or not. It can be seen that this is a binary classification problem.

For this project, we are given a dataset from the National Institutes of Health. This dataset includes a total of 27 558 cell images with equal instances of parasitized and uninfected cells,
We also have access to a starting kit where some features are already extracted, this is the dataset that we have mostly used.

What attracted us in this project, was the opportunity to learn about machine learning problems in the context of a real world application in medicine. We were able to see how what we learned was applied to medicine and how it could help in the process of diagnostic.

The work has been divided into 3 parts each taken care of by a duo. The three parts are in fact the classical steps/parts of a datascience project namely Preprocessing, Modelisation and Visualisation.

Classes' Description:

1)Preprocessing : RABIAI Samir et TRINH Duc-Bao

Preprocessing consists of preparing the data for the classifier, i.e. sorting, selecting and modifying the data. to allow the classifier to be more efficient.

We first started by doing some dimension reduction to see whether the datas were trivially separable, which would have allowed to get better result. But we've seen it wasn't.

Therefore we've decided to do some feature selection, in order to keep only relevant feature and not waste time on irrelevant/redundant feature. For this we've done two things :
_ First we've looked at the correlation matrix of our data and selected the five most absolutely correlated feature with the parasitized target. Giving us those five columns :

	min_gray	var_gray	nb_pixel_0.4_0.5	max_color	mean_gray	target
0	0.525490	0.472216	0	0.865132	0.595646	uninfected
1	0.623529	0.074386	0	0.959391	0.671309	uninfected
2	0.392157	0.658794	0	0.943739	0.611819	parasitized
3	0.568627	0.213046	0	0.879195	0.629835	parasitized
4	0.529412	0.060762	0	0.957338	0.678734	uninfected

Fig 1 – Head of our five most absolutely correlated features

_Afterward, we've also seen wich number of feature gives us the best score with a Kneighbors classifier for both our training set and test giving us those courbs :

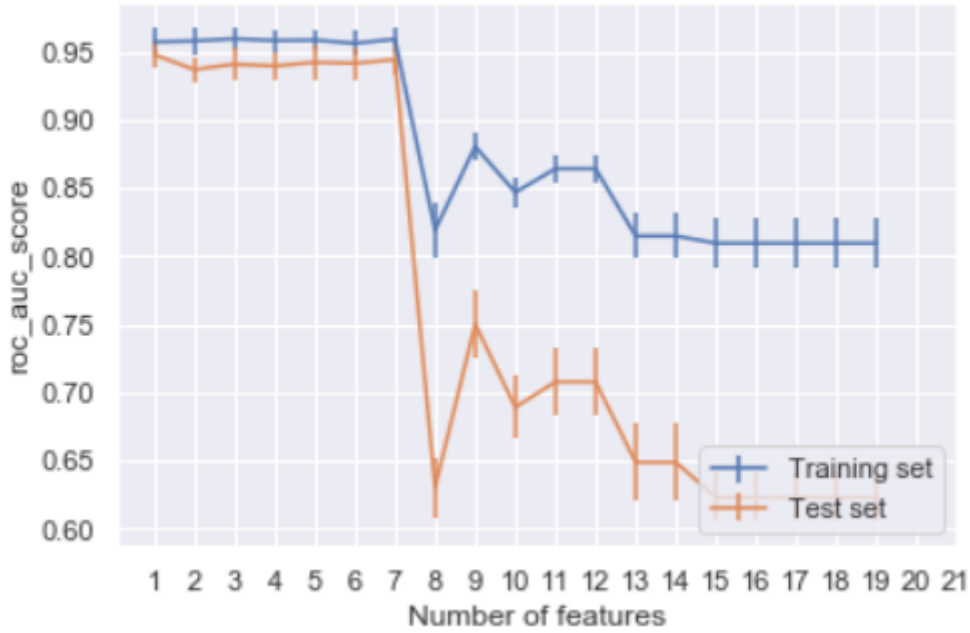

Fig 2 – ROC AUC Score depending on the numbers of features for both training and test set

Thanks to this we can see the fluctuation of the score is minimal for the 7 first features and the score itself is maximal for them too. As such we can simply reduce our dataset to those 7 features or even less given that the score stays approximately the same for the test set.

We've also looked for outliers, aberrant data points in our dataset. By deleting them, we could improve our score. For a small training set, we get the following result :

```
array([ 1, -1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1, -1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1, -1, -1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,
        1, -1,  1,  1, -1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1, -1, -1,
        1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1, -1,  1, -1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1,
        1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
       -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,  1,  1, -1,
        1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,
        1, -1, -1,  1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1, -1,  1,
       -1,  1,  1,  1,  1,  1,  1,  1, -1,  1, -1,  1,  1,  1,  1,  1, -1,
        1,  1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1])
```

Fig 3 – Visualization of the outliers on a small subset of our data

Where each 1 is an outlier. We can see that there is a non-negligible number of them.

2)Model : DJEBARRA Kahina et BOUAMARA Nabila

For the classifier, we retrieve the data that has just been processed. We will then compare several possible algorithms on training data and cross-validation to determine which one is best suited to classify our data. We will be able to choose the algorithm with the best result(s) on the values we are going to look at or well the algorithm among the best that remains the most constant.

The chosen models and their performance are :

	perf_tr	std_tr	perf_te	std_te
Nearest Neighbors	0.8	0.02	0.62	0.03
Linear SVM	0.93	0.01	0.92	0.02
RBF SVM	1	0	0.5	0
Gaussian Process	1	0	0.5	0
Decision Tree	1	0	0.93	0.02
Random Forest	0.99	0	0.94	0.01
Neural Net	0.67	0.08	0.65	0.07
AdaBoost	1	0	0.94	0.01
Naive Bayes	0.92	0.03	0.92	0.02
QDA	0.76	0.15	0.74	0.14

Fig 4 – Performance of the chosen Classifiers

3)Visualization : FELLAH Yanis et KOLCZYNSKI Samuel

The goal of the visualization duo is to allow a more intuitive understanding of our datas and to allow a better understanding of the happenings during the execution of an algorithm.

As such, we started by plotting our datas after dimension reduction, using mutiple methods, in order to see if we could distinguish 2 distincts clusters. If such was the case, we could have used the method which would have given us this result to latter obtain better results (and also diminish calculation time). Sadly it wasn't the case as can be seen in Fig 1.

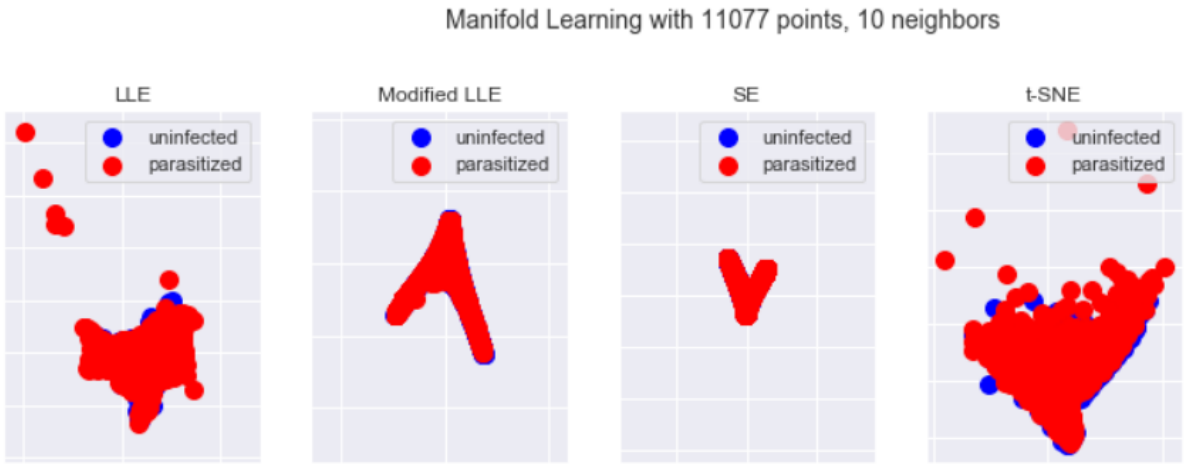

Fig 5 – Visualization of our datas after Dimension Reduction

Afterward, we plotted the correlation matrix of our dataset using a heatmap to see which component was correlated(or anti-correlated) with our target. We can see in Fig2 that the min gray component is heavily anti-correlated with our parasitized target. This plot can allow the prepocessing duo to more easily make choice when doing dimension reduction, as the more meaningfull a component is the more it influence our end result.

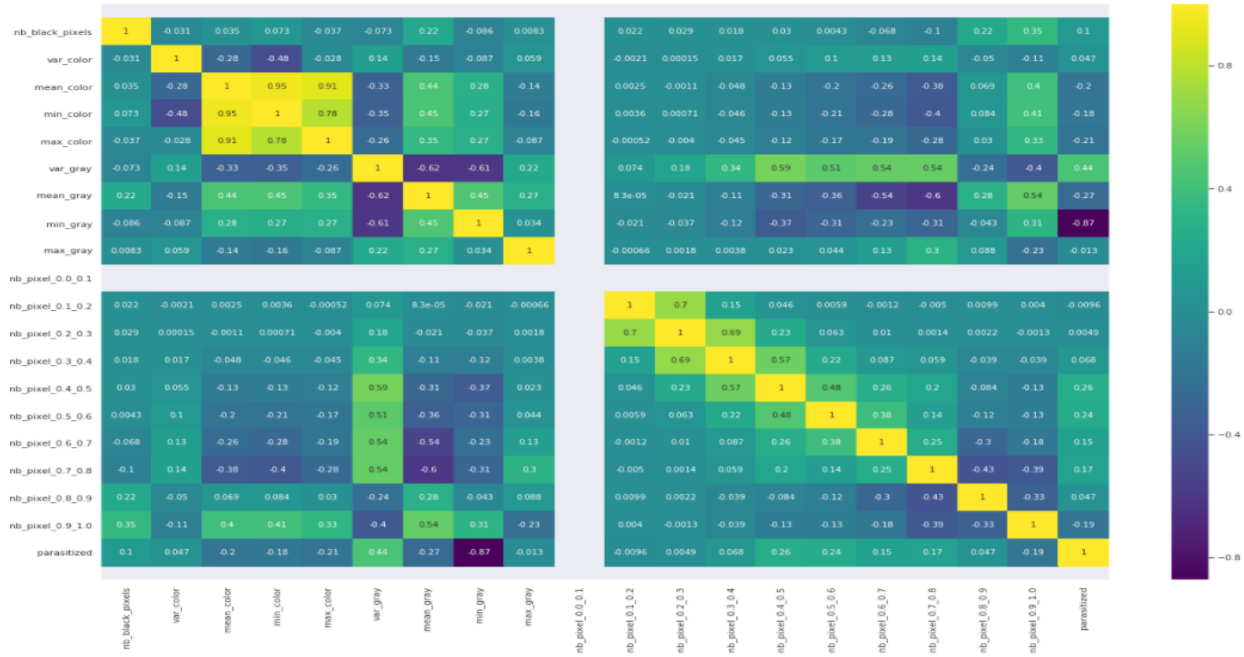

Fig 6 – Visualization of the correlation matrix using a heatmap

Afterward we've also done a visualization of the score of a model depending on the number of data points used in our training set accompanied with the variation of the score on the test set, this can be seen in Fig 3.

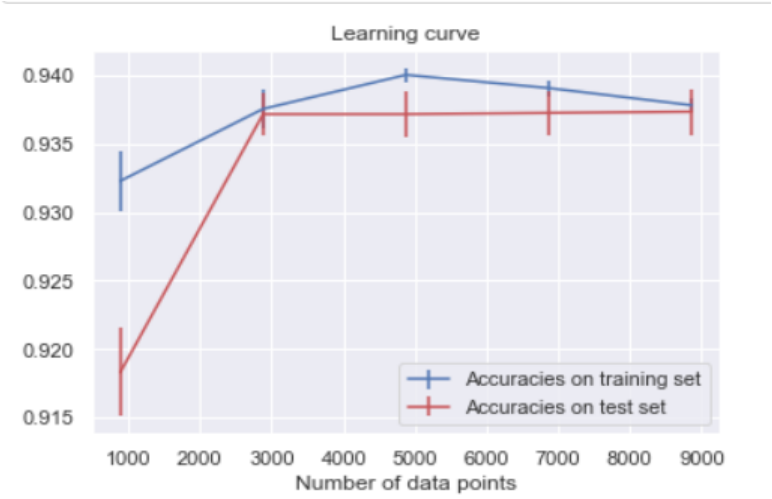

Fig 3 – Score depending on the number of points in our training set for both the training and test dataset

Finally, we've done a plotting of the decision boundary of a model which can be seen in Fig 4.

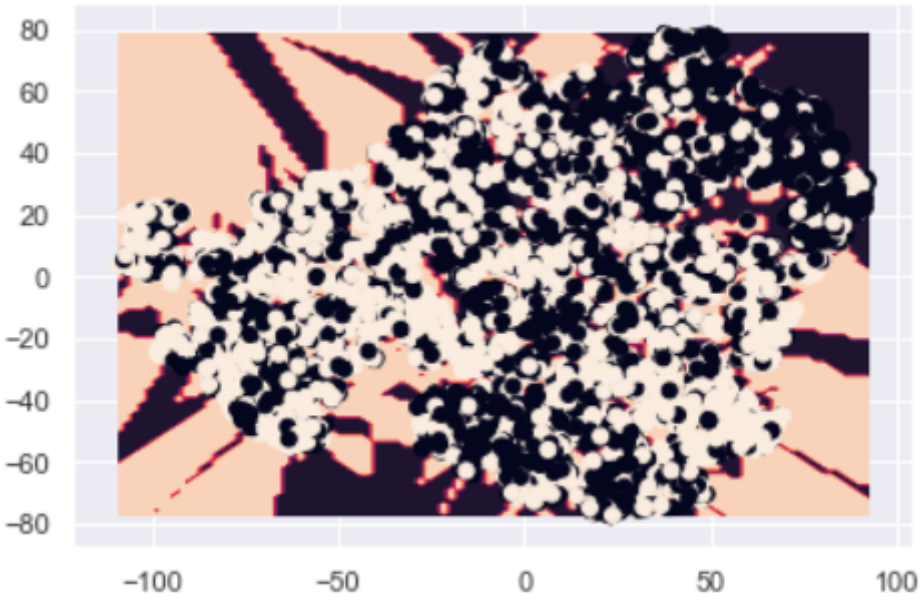

Fig 7 – Visualization of the decision boundary on our dataset (dimesion reduction applied)

References :

https://scikit-learn.org