LongRNA Classification for PedMS and ADHD: An Exploratory Approach

Luigi Minervini Federico De Laurentiis

15/01/2020

***Abstract* — The main goal of this study was to conduct exploratory work on the gene's expression of 47 pediatric patients, which can be healthy (HCPE), affected by multiple sclerosis (PedMS) or attention deficit hyperactivity disorder (ADHD). The aim is directed to discover some patterns and methods that can be used when working with data of genomic domain.**

It is in the interest of the experts to discriminate healthy patient from the unhealthy ones and, specifically, distinguish between ADHD and PedMS, which is a well-known challenging problem. In our work, we also found difficulties in the correct prediction of these two classes which seems to confirm this trend.

The study is organized around a pipeline composed by: Cross-validation, Over sampling, Feature Selection, Feature Scaling and Classification. For each phase, we explored and experimented with different methods and techniques, trying to optimize the performance of the architecture. Three different classification algorithms have been experimented with, namely Support Vector Machine, Neural Network and Random Forest. For each one, different combinations of feature selection, oversampling and scaling were used, as well as parameter tuning. In the end, we obtained good results with all the three classifiers, reaching the highest accuracy (0,76) with the Support Vector Machine using the LASSO and univariate feature selection method.

Keywords—computational intelligence, machine learning, neural networks, support vector machines, random forest, bioinformatics, LongRNA

I. INTRODUCTION

In this paper we are reporting the results of the analysis and classification, based on gene expressions, of pediatric patients which can be healthy (HCPE), affected by multiple sclerosis (PedMS) or attention deficit hyperactivity disorder (ADHD). These two conditions are both been shown to be correlated with the genomic expression of the subject. It is in the interest of the experts to correctly classify these three classes. More specifically, to correctly identify if a patient is affected by ADHD or PedMS.

Multiple sclerosis is a complex multifactorial disease of the CNS, characterized by inflammatory autoimmune demyelination and neurodegeneration. Usually identified as a disease of young adults, improvements in diagnostic tools and/or an increased sensitivity to its first signs and symptoms may have also contributed to a better recognition of MS during the very early ages of life, whereas in the past it had been mostly retrospectively tracked. At present, pediatric MS (PedMS) represents 3–5% of all MS cases.

Genetic predisposition, lifestyle, and environmental factors seem to contribute significantly to the overall risk of the onset of MS. Since the rarity of PedMS occurrence, very few studies have investigated the genetic involvement of PedMS. Attention Deficit Hyperactivity Disorder (ADHD) is a childhood-onset neurodevelopmental disorder characterized by inappropriate and impairing inattention, impulsivity, and hyperactivity. The disease occurs in 2%–10% of school-age children, more frequently identified in young males. ADHD is considered a complex disorder caused by environmental, epigenetic, and genetic factors. Several candidate genes have been implicated in ADHD susceptibility.

The goal of our work is to create a predictive model which is able to distinguish healthy patients from the affected ones, in particular distinguish PedMS from ADHD subjects.

The data, used to train the models, has been collected using the Next Generation Sequencing method which provides an inexpensive, genome-wide sequence readout. The dataset, provided by the CNR Bari, represents the expression of each gene in the patient. The data results very unbalanced, therefore it will require the application of some oversampling techniques. Due to the particular method used for the reading (NGS), it also has a high dimensionality of gene expressions and will require some sort of feature selection.

Our approach was exploratory, for each step (oversampling, feature selection, feature scaling and classification algorithm) we tried different approaches and algorithms. The aim was directed to best results of the model on accuracy, precision and recall metrics among the three classes. From the set of methods performed we select and discuss the best scoring ones in comparison with the worst ones.

II. RELATED WORKS

Next generation sequencing (NGS) technologies provide a high-throughput means to generate a large amount of sequence data. Further, highly efficient and fast processing tools are required to handle the large volume of datasets [4]. The dataset under study was created using this technique on samples of pediatric patients.

Multiple sclerosis is a CNS inflammatory demyelinating disorder that includes both self-limited and lifelong conditions, which can be indistinguishable at the time of initial presentation [5]. The disease onset typically occurs in young adults, especially females, although diagnosis during childhood and adolescence has been increasingly recognized worldwide, accounting for 3−10% of the whole MS population, hence called pediatric MS (PedMS).

Given the complex heterogeneity of MS, the possibility to identify reliable markers, predictive of the clinical course or response to treatments, has so far been elusive due to its multifactorial nature that involves several genes and their interactions. The symptoms are unpredictable. Some people's symptoms develop and worsen steadily over time ("relapses"), while for others they come and go ("remissions"). Most people with MS only show few of the related symptoms. Circulating markers (miRNAs), target genes (mRNAs) and functional pathways associated with PedMS have been identified by the authors of [2].

ADHD is a complex brain-based disorder that includes both neurocognitive and behavioral difficulties such as inattention, hyperactivity, impulsivity, and challenges in planning, organization, self-evaluation, and mood stability. ADHD-related symptoms manifest in impairments in children's academic and behavioral functioning, as well as in their social interactions (such as play) [6]. It is considered a childhood-onset neurodevelopmental disorder. Given the clinical heterogeneity of ADHD and its high comorbidity with other psychopathological disorders, there is an urgent need to identify useful molecular signatures that help in deciphering the disease pathogenic mechanisms, thus facilitating the diagnosis and possibly addressing new therapeutic strategies. Some novel regulatory networks and molecular pathways possibly related to ADHD had been found in by [1].

The dataset analyzed in this study contains both samples of PedMS and ADHD and it has not been analyzed yet; therefore, not much information about possible pre-existing model has been found. However some classification attempt with gene-type data has been tried with a Random Forest approach [3].

Most of our work is centered on the analysis of the dataset using different Machine learning techniques such as: Cross-Validation, Over sampling, Feature Selection and others. Most of the algorithms used were taken from the documentation page of sklearn which provide a rich and detailed descriptions of the methods as well as an example of their usage.

III. Materials and Methods

In this section will describe the data and the methods used to obtain the final prediction models and to benchmark the various configurations. In particular:

- **Data:** The dataset, "longRNA_NGS", is composed by 47 row and 37058 columns. It has been provided from the ***CNR of Bari.*** Each row represents a patient which can be: healthy (HCPE), affected by pediatric multiple sclerosis (PedMS) or affected by attention deficit hyperactivity disorder (ADHD). The data has been collected using the Next Generation Sequencing (NGS) technique which provides a genome readout from samples of 47 different pediatric patients. A column represents a specific gene associated with a patient; every value is an expression of that specific gene for that patient. All data is of numerical, non-negative type, and no missing value has been found. Before starting with the analysis, some preprocessing was necessary in order to reduce the dimensionality of the dataset and enhance the performance of the various classification algorithms used in the process.
- **Methods**: Our work is organized in a pipeline composed by following steps:
  - **Cross-Validation**
  - **Over sampling**
  - **Feature Selection**
  - **Feature Scaling**
  - **Classification**

A. Data

As previously stated, the dataset describes collected samples of genes from 47 pediatric patients which are divided into three classes:

- Affected by Pediatric Multiple Sclerosis i.e. PedMS (19 samples)
- Affected by affected by Attention Deficit Hyperactivity Disorder i.e. ADHD (8 samples)
- Healthy children i.e. HCPE (20 samples)

For each patient, a total of 37058 features are collected. All the values represent the level of expression of a gene in the patient, defined as the frequency of RNA copies produced from the DNA: each gene can produce different RNAs and both strands of DNA produce RNAs. According to the literature, the higher the value, the higher is the expression of that gene in the patient.

Is important to say that the raw data is presented to us in a very unbalanced form, the number of features is much more grater to the number of samples and the number of patients in each class is not constant within them (especially for the ADHD class).

For a better understanding of the distribution involved in the data some simple statistical analysis has been performed in order to better adjust the type of preprocessing to be done next.

The dataset is composed by numerical-type, non-negative values except for the class labels which are of categorical type.

The mean value in the dataset is around 490 but is important to say that most of the average values settle around 0, with a small set of features that has a very high value. However, sometimes a mean can be misleading and may not effectively show a typical value in our set of data. A common method to measure the variation of our values is to calculate the standard deviation (SD). The SD is a measurement to tell how a set of values spread out from their mean. A low SD shows that the values are close to the mean while a high SD shows a high diversion. The mean SD value in the dataset is around 542 but similarly to the mean most of the SD values on the features are very low, with a small set of features that has a very high value.

In both cases (means and SDs) there are three spike values associated with the same genes, specifically: 'gene_26248', 'gene_16092', 'gene_18011'.

Finally, to give some description of the range of values we are working with, we calculated:

- the minimum value which is 0
- the maximum value which is 61544818.5
- the mean minimum value which is 136.3
- the mean maximum value which is 3275.4

B. Methods

In order to better attack the problem, all the methods used to analyze the dataset, have been organized in a pipeline. For each step several approaches are tried, searching for the best configuration of methods and their parameters. In the following sections we will show the different methods used, with a brief description.

1. Cross-validation

Cross-validation is a statistical method used to estimate the how good a machine learning model will generalize to an independent dataset. The aim of this step is to divide the set of data in the train and test sets. Two main methods have been implemented:

- **Stratified K-Folds cross-validation**. The splitting of data into folds is governed by the criteria, according to which, each fold has the same proportion of observations with a given class outcome value. This technique was chosen due to the unbalanced nature of the data under study, to make sure that each class was represented in our test samples.
- **Leave One Out**, using this technique, each learning set is created by taking all the samples except one, the test set being the sample left out. This cross-validation procedure does not waste much data as only one sample is removed from the training set. However, the LOO has a high variance which often makes it a bad choice of estimator for performance evaluation, even though it is approximately unbiased.

2. Over sampling

One of the difficulties posed by the dataset under study is the inherit unbalance between the three classes. In fact PedMS, HCPE and ADHD have disproportionate samples with respectively 19, 20 and 8 entries. This type of data may have negative repercussions on the final prediction of the classification model, adding a bias towards the class with the higher number of samples. To mitigate this problem, we made use of oversampling techniques.
Oversampling involves introducing a bias to select more samples from one class than from another, to compensate for an imbalance which, in our case, was an under-representation of the ADHD class. The end-result of this technique is the creation of a balanced dataset. Many machine-learning algorithms, such as neural networks, make more reliable predictions being trained with balanced data. Certain analytical methods, however, may not benefit from a balancing approach. Two oversampling techniques have been tried:

- **Random Oversampling** involves supplementing the training data with multiple copies of some of the minority classes. This is one of the earliest proposed methods, that is also proven to be robust. Instead of duplicating every sample in the minority class, some of them may be randomly chosen with replacement.
- **Synthetic Minority Over-sampling Technique (SMOTE)** takes a sample from the dataset and considers its k nearest neighbors (in the feature space). To create a synthetic data point, the vector between one of those k neighbors, and the current data point are taken. Then the vector is multiplied by a random number x which lies between 0, and 1 and added to the current data points to create the new, synthetic sample.

3. Feature Selection

Another challenge the data presented was the high dimensionality in the features dimension. This is an inherited cause of the technique used to obtain the gene readouts, the NGS, which identifies **all** the RNA content of a biological sample.
It then proved necessary to implement some techniques of feature selection in order to reduce the dimensionality, trying to remove only the redundant and superfluous features. This is done for two reasons: selecting a significant subsample of features, may help with the prediction because the classifier might focus more on the statistically significant features; the other reason is to make some calculations more feasible, reducing the computational cost.
Following we will discuss the feature selection method implemented, with a brief description:

- **Low mean elimination**, this method aims to remove all the features that have an average in the samples less than some specified threshold. Considering that the dataset is composed by non-negative values, we want to discard features that have a mean that is very close to zero.
- **Low variance elimination**. Similarly, to the method above, we want to remove only the features that express a variance smaller than some threshold. This will remove all non-zero variance features (essentially that have the same value in every sample) and, depending on the threshold, genes that have very similar values in every patient (non-discriminant).
- **Univariate feature selection**, this method works by selecting the best features based on univariate statistical tests, comparing each feature to the target variable to understand whether there is any statistically significant relationship between them. When we analyze the relationship between one feature and the target variable, we ignore the other features. The scores are compared, and the best scoring features are selected. As a parameter we will specify how many features we want to extract (k) and the statistical test to be applied (e.g. chi2).
- **Decision tree feature selection**. Decision Trees are a non-parametric supervised learning methods used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. Once trained on all the features, the importance of a feature is computed as the (normalized) total reduction of the criterion brought by that feature. It is also known as the Gini importance. As a parameter we will specify how many features we want to extract (k).

- **Least Absolute Shrinkage and Selection Operator (LASSO)** is a powerful method that perform two main tasks: regularization and feature selection. The LASSO method puts a constraint on the sum of the absolute values of the model parameters, the sum has to be less than a fixed value (upper bound). In order to do so, the method apply a shrinking (regularization) process where it penalizes the coefficients of the regression variables, shrinking some of them to zero. During the feature selection process, the variables that still have a non-zero coefficient after the shrinking process are selected to be part of the model. This method extracts a non-specifiable number of features ranked on their importance in the model.
- **Correlation matrix feature elimination**. In this method we build a correlation matrix and select only the features with low correlation between them. The matrix contains the correlation coefficient between two variables. This is a value between 1 and -1 and refers to how close two variables are to having a linear relationship with each other: a value near zero indicates no correlation, while a value closer to ±1 indicates either a positive or negative correlation. If two variables are strictly correlated, it will not bring much more information to the predictive model to include both. As a parameter we will specify a threshold below which features will be dropped. The aim is to eliminate variables close to ±1 and only keep one of them. This method is computationally demanding as it creates a matrix $n \; x \; n$(with n as the number of features) so it will be infeasible to use feature elimination based on correlation alone. This method will require feature reduction with another algorithm before applied.
- **Recursive feature elimination (with random forest classifier)**. Random forest (RF) is a machine-learning method that generally works well with high-dimensional problems and allows for nonlinear relationships between predictors; however, the presence of correlated predictors has been shown to impact its ability to identify strong predictors. The Random Forest-Recursive Feature Elimination algorithm (RF-RFE) mitigates this problem in smaller data sets, but this approach has not been tested in high-dimensional genomics data sets. This method is computationally taxing so, like the above it will need feature reduction before its application.

4. Feature Scaling

The next logical step in our preprocessing pipeline is to scale our features. The aim is about making sure that data is internally consistent, that is, each data value is in the same scale and comparable with each other. Two feature scaling methods have been implemented:

- **Standardization** is a transformation that centers the data by removing the mean value of each feature and then scale it by dividing (non-constant) features by their standard deviation. After standardizing data, the mean will be zero and the standard deviation one. Standardization can drastically improve the performance of models, for instance, the RBF kernel of Support Vector Machines highly benefits from standardization. If a feature has a variance that is orders of magnitude larger than others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected.
- **Minmax Scaler** transforms features by scaling each feature to a given range. This range can be set by specifying the feature range parameter (default at [0,1]). This scaler works better for cases where the distribution is not Gaussian, or the standard deviation is very small. However, it is sensitive to outliers.

5. Classification

Finally, given the application of the previous steps, we want to train a classifier in order to obtain a prediction model that can be evaluated on the test set. Note that previous steps are applied to the training set (oversampling, feature selection and feature scaling) and are conducted covertly to the test set in order to avoid a bias that would inevitably lead to optimistic performances. However, the same transformations are applied to the test set to obtain comparable data.
In particular, we are using supervised learning methods which require previously classified reference samples in order to train the classifier and subsequently classify unknown data. The target classes are: HCPE (class 0), ADHD (class 1) and PedMS (class 2). The classification methods implemented are:

- **Support Vector Machines (SVM).** An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. There are many hyperplanes that might classify the data. One reasonable choice as the best hyperplane is the one that represents the largest separation, or margin, between the two classes. So, we choose the hyperplane so that the distance from it to the nearest data point on each side is maximized. As a parameter this method takes a kernel function to apply in the classification.
- **Random Forest Classifier (RFC).** As already discussed, RFC is an ensemble learning method for classification, regression and other tasks. It operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set. As a parameter this method takes the decision trees' maximum depth allowed.
- **Feed-Forward Artificial Neural Networks (ANN).** This model consists of multiple layers of

computational units, interconnected in a feed-forward way. Each neuron in one layer has directed connections to the neurons of the subsequent layer. The units of these networks apply an activation function before the output to the next layer. Multi-layer networks use a variety of learning techniques, the most popular being back-propagation. Here, the output values are compared with the correct answer to compute the value of some predefined error-function. By various techniques, the error is then fed back through the network. Using this information, the algorithm adjusts the weights of each connection in order to reduce the value of the error function by some small amount. After repeating this process for a sufficiently large number of training cycles (epochs), the network will usually converge to some state, where the error of the calculations is small. To adjust weights properly, an optimization function is applied. More in depth, our Net, is composed by two layers of 32 neurons, fully connected, with a ReLu activation function, and an output layer with a sigmoid activation function; moreover, we used the categorical cross-entropy as loss function and stochastic gradient descent as the optimization function. This configuration was kept after some trial and error process. We kept as a parameter only the number of epochs on which the Net will be trained.

IV. EXPERIMENTS

As already described, this work is centered around a "pipeline". With this term we are referring to the application of different steps in order to obtain a predictive model to test and evaluate. The raw data is the input of the process and transformed between each step. As the output we have the performance metrics of a classification algorithm trained on the processed data. More in depth the pipeline will be of the type:

Cross-Validation ->
Over sampling (on train set) ->
(One or more) Feature Selection method(s) (on the train set separately from the test set) ->
Feature Scaling (on test and train sets separately) ->
Classification ->
Evaluation

In order to understand the best method combinations and parameters, several experiments were performed.
We define an "experiment" as a configuration of the pipeline with specific methods, parameters and different distributions of test and training sets.
The metrics used to compare the different models are precision, recall and accuracy above the three classes.
In machine learning, the accuracy is defined as the ratio between the number of correct predictions over the total number of predictions. Precision is defined as the proportion of positive identifications that were correct, while recall (also known as sensitivity) is the proportion of actual positives that were identified correctly. Even considering all these three metrics we focused on the maximization of the general accuracy of the system.

The programming language used for the implementations is python, with the help of several libraries such as pandas, NumPy and sklearn.
The organization of the experiment, and in general of our work, have been a key factor for a good analysis. Each method, before being applied, was tested on Jupyter notebook, in order to discover its potential and the best approach to be used. Gained enough experience with a method, we implemented a script to be added in the system and create new experiments with it, integrating it in the pipeline.
An important tool which helped in the evaluation process has been MLflow which is an open source platform to manage the Machine Learning lifecycle, including experimentation, reproducibility and deployment. The tracking option was used, which tracks the metrics achieved in each experiment in relation of the parameter used. This made easier the evaluation of each experiment and helped to create replicable result.
The study is focused on the evaluation of different combinations of approaches and techniques. The results of each experiment guided our work and allowed us to make decisions for the next.
Each method can have multiple parameters to be optimized (specifically for the data under study). Each time different algorithms are composed together it's not obvious that the best parameters found for a method in some configuration gives similar results with a different one. This makes more difficult to find the best architecture possible.
The number of feasible pipelines (in relation to the possible setting for each method) increases dramatically with all the possible combinations of method. In order to attack the problem, some opportunistic decisions, guided on the result achieved, were made. Some methods described in the section above, although implemented, were excluded to reduce complexity and remove some free variables.
First, we tried to discard some methods based on informed decisions. We chose to apply only SMOTE algorithm as oversampling algorithm. Due to the small size of the dataset, Random Sampling would create copies of samples in the minority class (ADHD), resulting in a not enough generalize models.
Likewise, for feature scaling we considered only the Standardization method at the expense of Minmax scaling. This choice was motivated by the fact that, in genomic expression, the values are expressions of the manifestation of the gene. The use of Minmax may eliminate this information, scaling all the values in a fixed interval (between 0 and 1 by default).
Finally, we excluded the Leave One Out method in favor of Stratified K-Fold validation. Although LOO would probably give more accurate estimates it would be much more time consuming. With a lot of different combinations of pipelines to try, our choice was oriented to the stratified K-Fold cross validation.
After having eliminated some methods, with some informed guesses, and having reduced the number of different possible experiments to evaluate, we focused on feature selection. Some experiments were performed to remove possible combinations of feature selection algorithm at this step of the pipeline. More in depth we found that:

- The performances do not change using variance elimination before applying Univariate selection or decision tree selection. Therefore, we can safely exclude these configurations.

- The number of output features of the LASSO method does not change significantly using low variance selection before. We can then exclude this composition.

- The correlation and the recursive feature selection methods cannot be used on the initial data because the computation cost is too high. We can then ignore the use of these two algorithms before any other one.

Finally, having restricted the pool methods, we started experimenting. For each classification algorithm, first we used a single feature selection and then, we composed them using (as a guide) the parameters obtained from previous tuning. We repeated this, in combination with the application (and also non-application) of: SMOTE oversampling, Standardization feature scaling and different sizes of fold for the Stratified cross validation method (namely 3, 5 and 7). The most interesting results will be reported and discussed in the following section.

V. RESULTS

In this section, we will discuss the best performing models and their associated experiment pipeline, as well as the worst performing ones, in order to understand what can be improved. We will talk about the best configuration obtained for each classification algorithm used and then discuss some final considerations.

Support Vector Machines

Some of the best results that we achieved during the analysis, are obtained with the use of the SVM classification algorithm. This classification method scored the highest with the use of oversampling and standardization. Interestingly, this approach was also the most stable and had a consistent performance across all the experiments. In addition, no substantial fluctuation of score, in relation to the different possible kernel tried, was noticed. Starting from these results, every experiment performed with the SVM algorithm, used a radial basis function (rbf) kernel (default in the sklearn library).

For this predictive model, we notice that using only one feature selection method is enough to achieve good results. With the univariate feature selection (with k=1000) the SVMs achieved an accuracy of 0,7 (Exp.1 in Table I).

In general, across all experiments, the SVMs achieved a general good performance with the higher accuracy reached combining univariate selection and LASSO feature elimination. This score follows the trend noticed on the single feature selection approach and is the best result that we achieved yet (accuracy of 0,76, Exp.2 in Table I).

From our results, the SVM algorithm seems the most reliable and consistent approach, obtaining very similar results across folds.

Neural Networks

The Neural Network (NN) used, as mentioned before, is composed by two, fully connected layer, composed by 32 neurons with a ReLu activation function while the output layer uses a sigmoid activation function. We used this fixed architecture experimenting with only the number of epochs; starting from 5 to 100, with a step of 10. Empirically the best accuracy is reached with 40 epochs. This value was then used for all the other experiment with the NN.

The best results for this classifier are achieved with the use of both oversampling and standardization.

For the features selection methods used, by itself, the low variance elimination reached an acceptable accuracy (0,7 Exp.4 in Table I) using a threshold of 0,8.

Interestingly, mixing univariate feature selection (k=500) and LASSO, we reached 0,66 accuracy score (Exp.5 in Table I); slightly less than the use of low variance elimination method alone. The decreasing of performance, may be due to reduction of features used when training the network.

In general NN tends to have inconsistent performance across folds, in comparison with the other methods (specifically to the SVM). This may be because the initial state of the network (weight of the nodes) start with a random value which can so vary each time the network is trained.

Random Forest

The last classification algorithm considered is the Random forest algorithm (RF).

First we chose the depth of the tree, testing with different values (1, 3, 5, 7, 10, 20, 30) and noticed best performances reached with a depth of 3. This parameter was then used as a base of all the next experiments.

In general, the RF approach, achieved less accuracy compared with SVM and NN. For feature selection, using just the low variance elimination method, we reached an accuracy of 0,64 with both SMOTE oversampling and Standardization scaling (Exp.6 in Table I).

Mixing different feature selection methods, we obtained the highest accuracy (0,68 Exp.7 in Table I) with the use of decision trees (k=500) and correlation algorithms for removing features. The latter is the best scoring result for this classifier, obtained with only the use of oversampling but without standardization.

We observed that, even if the general accuracy within the RF experiments is less compared with NN and SVM, its consistency is comparable with the SVM.

All the results above are reported from experiments performed with a 3-fold cross validation. Note that while experimenting with higher folds, a slight improvement in the performances was noticed (e.g. with 5 and 7 folds). This result might be correlated with the greater availability of training data.

In addition, experiments without the use of standardization tend to perform worse with respect to the use of feature scaling. One exception is the Random Forest Classification as we discussed above. Notably, the Neural Network seems to benefit the most from standardizing data, having the worst performance, with every algorithm of feature selection (as an example see Exp.9 in Table I).

Finally, even among the best scoring models described above, we found some difficulties in the correct classification of the ADHD class, which is notably the less populated, with

a total of 8 samples (the best precision for ADHD class is of 0.57). In fact, across all the results we find that all the classifiers can correctly distinguish between healthy and unhealthy patients. In fact with the same methods we tend to reach an accuracy of 0.91 just distinguish between these two classes. The challenging task is to correctly distinguish PedMS and ADHD.

To mitigate this problem, we tried to combine all the classifiers with ensemble learning to obtain better prediction performances. Specifically, hard majority vote across the outputs of the three classifiers was used, combining the outputs into one final prediction. Note that the same Oversampling, Feature Selection and Standardization was used for all three models, obtaining the best accuracy of 0.77 and a precision on ADHD class of 0.6 (Exp. 12 in Table I).

As a last experiment, we followed again a hard majority vote approach, but, this time, training each classification algorithm with the best preprocessing pipeline for that specific method. In this context we reached the accuracies up to 0.83 (with precision on ADHD class of 0.71) which is the best outcome yet (Exp.14 in Table I).

In Table I are collected the most interesting experiments.

VI. CONCLUSIONS

Summarizing, our results may show some best patterns and methods (among the analyzed ones) that can be used when working with data of this domain. Facing the problems this type data brings, we obtained some generally good results.

Although the metrics achieved seem promising, all the experiments faced problems in the identification of patients with Attention Deficit Hyperactivity Disorder. This might be connected to the dimensionality of the samples for that class or because of the similarity of in feature expression with the PedMS class. All the models, in fact, tend to wrongly classify between the two types of unhealthy patients. To solve this problem, we used a method of ensemble learning that seems to achieve better results. A future development in this respect may be to use different methods to combine predictions i.e. to weight each guess from a classifier based on their general accuracy or precision for a given class.

Another improvement may be achieved using new data (not genomic) that correlates to the same patients, in order to have other variables that might result discriminant for the MS and ADHD conditions.

VII. REFERENCE

[1] N. Nuzziello, F. Craig, M. Simone, A. Consiglio, F. Licciulli, L. Margari, G. Grillo, S. Liuni e M. Liguori, «Integrated Analysis of microRNA and mRNA Expression Profiles: An Attempt to Disentangle the Complex Interaction Network in Attention Deficit Hyperactivity Disorder,» *Brain Sci*, 2019.

[2] L. Maria, N. Nuzziello, F. Licciulli, A. Consiglio, M. Simone, R. Viterbo, T. Creanza, N. Ancona, C. Tortorella, L. Margari, G. Grillo, P. Giordano, S. Liuni e M. Trojano, «Combined microRNAs and mRNAs expression analysis in Pediatric Multiple Sclerosis: an integrated approach to uncover novel pathogenic mechanisms of the disease,» *Human Molecular Genetics*, 2017.

[3] B. Darst, M. K. e C. Engelman, «Using recursive feature elimination in random forest to account for correlated variables in high dimensional data» *BMC Genet*, pp. 19-65, 2018.

[4] P. RK e J. M, « NGS QC Toolkit: A Toolkit for Quality Control of Next Generation Sequencing Data» *PLOS ONE*, 2012.

[5] L. B. Krupp, B. Banwell e S. Tenembaum, «Consensus definitions proposed for pediatric multiple sclerosis and related disorders» *Neurology*, Apr 2007.

[6] L. A. Reddy e A. Alperin, Handbook of Child and Adolescent Group Therapy, 2012.

exp	oversampling	feature selection	scaling	classifier	accuracy	precision	recall	Kfold
1	SMOTE	univariate (k=1000)	normalization	SVM	0,7	0,66	0,64	3
2	SMOTE	univariate (k=1000) + lasso	normalization	SVM	0,76	0,7	0,67	3
3	SMOTE	univariate (k=1000) + recursive	normalization	SVM	0,74	0,77	0,67	3
4	SMOTE	low variance (0.8)	normalization	NN	0,7	0,6	0,62	3
5	SMOTE	univariate (k=500) + lasso	standardization	NN	0,66	0,43	0,44	3
6	SMOTE	low variance (0.8)	standardization	RandomForest	0,64	0,58	0,61	3
7	SMOTE	decision tree (k=500) + correlation (0.8)	None	RandomForest	0,68	0,57	0,57	3
8	SMOTE	decision tree (k=1000) + correlation (0.8)	None	SVM	0,3	0,39	0,41	3
9	SMOTE	univariate (k=500) + recursive	None	NN	0,3	0,14	0,33	3
10	SMOTE	decision tree (k=500) + correlation (0.8)	None	RandomForest	0,79	0,72	0,75	5
11	SMOTE	low variance (20)	standardization	NN	0,75	0,68	0,73	5
12	SMOTE	low mean (0.8)	standardization	hard majority vote	0,77	0,74	0,75	5
13	SMOTE	low variance (0.8)	standardization	hard majority vote	0,76	0,73	0,77	7
14	SMOTE (for all classifier)	univariate (k=1000) + lasso (SVM) decision tree (k=500) + correlation (0.8) (RF) low variance (20)(NN)	standardization (SVM and NN none for RF)	hard majority vote	0,83	0,79	0,82	7

Table I. List of significant experiments.