Assignment 2: The Raven Test
In Search for Group Differences

Filipe Pires[1]
85122

João Alegria[1]
85048

Ana Tomé[12]
ana@ua.pt

[1] Department of Electronics, Telecommunications and Informatics,
University of Aveiro, Portugal

[2] Institute of Electronics and Informatics Engineering of Aveiro,
University of Aveiro, Portugal

Abstract

The search for trustworthy methodologies of determining a measurable intelligence index has been a quest of our brightest minds for decades. Raven matrices tests have proved to have widespread practical use as a measure of intelligence. They are a source of data for many studies on the general population as they seem promising tools for contexts such as psychometric tests or clinical assessment.

In one study on the application of these matrices to groups of students from different backgrounds, questions emerged regarding the possibility of clear differences between Multimedia and Informatics students. In this paper we present the statistical analysis applied to the tests results with the help of ML classification techniques in search for determining whether any of the two groups showed significant advantages over the other.

1 Introduction

Raven's Advanced Progressive Matrices (RAPM) is a non-verbal group test typically present in educational or clinical settings, as it is used in measuring abstract reasoning and regarded as an estimate of fluid intelligence [1]. Examples of related test are Naglieri Nonverbal Ability or Spacial Ability Tests. Their practical use is very extended, and applicable to both adults and children. Nevertheless, studies that resort to them usually focus on populations containing groups with specific differences in order to draw conclusions from these differences. Examples of these studies are on different military sections, or different mental disabilities.

In the study whose collected data was used for our analysis [2], the aim was to compare students from different fields in terms of learning styles effectiveness. Several tests such as Kolb and VAK or Hermann dominances allow to distinguish some learning styles like: Accommodator, Assimilator, Auditory, Convergent, Divergent, Kinesthetic and Visual. But beyond this, the researchers also applied the RAPM tests to reach more robust conclusions, and combine all results in a meaningful way. In this paper we focus only on the data related to the second set of tests.

The population that performed the Raven tests was a group of 45 university students, 21 of Design and Multimedia and 24 of Informatics Engineering. 48 problems were presented to both populations, divided into 2 phases: during the first 12, the participant would receive a feedback about his/her answer; for the remaining 36 no feedback was given. During test execution, electroencephalographic (EEG) signals were registered while participants performed the tasks, using Enobio 8 EEG recording headset and 8 channels: *F3, F4, T7, C3, Cz, C4, T8* and *Pz*.

Our aim was to determine from this estimated measurement of intelligence whether both groups hold characteristics significantly different from each other by building classification algorithms that interpret the EEG signals and other time-related metrics as features and attempt to predict which class of students a new entry belongs to. We also intended to compare our conclusions with those obtained by the original researchers.

2 Dataset & Feature Extraction

The dataset provided for this assignment had data both related to individual participants and aggregations with averages. For our intentions, we were interested only in the individual results, so we used the data from the "Trial" folder, containing 4 XML spreadsheets: two for the Informatics students (referred to as DEI), two for the Multimedia students (referred to as ESEC). Departments were split as one file contains the right answers of the students on the Raven tests and the other contains the wrong answers.

Each file contains information about 36 task executions, with an entry for each student and a column for each metric. The relevant marks for the signal analysis are around:

- Problem Display
- Possible Solutions Display
- Student's Answer

On the problem and possible solutions displays, the considered signal processing windows were [-75 500] ms. On the student's answer, the window was [-500 500] ms.

Some of the considered metrics were:

- Peak & Latency for P100 PZ, P100 CZ, P300 CZ signals
- F3, F4 Stress signals

P100 and P300 were chosen because of attentional and relationship characteristics that have sometimes been attributed to them. They are defined in the first time interval and their latency and amplitudes are stored, considering only the relevant channels. This data was used as time features for the classification algorithms.

For the frequency features, the energy of the characteristic bands is estimated in all defined windows. This Engergy (E) is then used to compute other higher level features called energy ratios, considering several signal channels:

- Stress Index
- Mental Fatigue
- Alpha Lateralization
- Immersion Index

All of these features are present in the dataset spreadsheets.

As in any classification challenge, we needed to divide the data into training and testing entries. The way we divided the results was 80% for the training of the classifiers and 20% for testing, due to the relatively reduced size of the dataset.

3 Data Quality & Normalization

The first thing we did with the raw data was to correlate every possible pair of features through 2-Dimensional projections. This would help us understand how was the data distributed in the multidimensional space and whether there were clear separations between the two classes (DEI and ESEC). Table 1 is the visual representation of this correlation as an organized collection of:

- Histograms - presented in the diagonal composed by the feature related with itself and describing the value distribution inside that given feature.
- Scatter Plots - presented in all other entries and describing the value distribution between feature pairs.

Through the analysis of this table, we can gather some important conclusions. The most clear characteristic detected was the presence of outliers, one that was already expected and needed to be taken care of. The second was the fact that the data in all pairs of features doesn't appear to be separable, i.e. there should be something resembling two groups, one for each class, or point clouds in some of the projections. If this was visible, the possibility of actually existing a difference between groups would be likelier to be true. However, that does not seem to be the case, since in most cases we are presented with uniform distributions, gaussian distributions or simply the grouping of all data in a unique central point. It is important to mention that this does not automatically mean that a solution could not be found, as the group distinction we were looking for could depend on more than two variables and so could indeed not be visible in the 2-Dimensional projections.

Once familiarized with the data, we proceeded to pre-processing it so that the models we intended to to apply would have the best chances of finding the distinguishing factors. To do this, we centered the data and removed the detected outliers. Equation 1 contains the formula of the *Z-Score* algorithm applied to each feature to center all values. Then, we removed any data entry that contained a *Null* value, since that would cause problems for the models' processing. Finally we removed the still existing outliers with the help of equation 2, that ensures that entries with very distant values from the standard deviation are not considered as they do not have valuable information to offer.

$$z_i = \frac{x_i - \mu}{\delta} \tag{1}$$

$$|x_i - \mu| < 3 \times \delta \tag{2}$$

4 Classifiers

In this section we will present and give a brief summary of all the machine learning models used in this study. We chose this models since through a brief state f the art analysis we concluded that these are the most common and suggested models when handling machine learning problems, and also the ones that achieve the best performance overall. We also found that some of the models used are to most recommended when handling with data similar to our context.

4.1 Support Vector Machine

Support Vector Machine, commonly denoted as SVM, is a supervised machine learning model that with the supplied training set, maps the different data entries into a multidimensional space in a way that the widest possible gap exists between the data of each class. Based in that gap a model is created and a decision rule defined that when feed new data tries to assign a class to that entry by comparing against the decision rule already mentioned.

The most common and simple SVM application tries to creates a linear division between the presented classes. Another alternative that allows more complex decision rules is the application of a kernel trick, that implicitly maps tha data to a higher-dimensioned feature space which may enable a better separation of the data.

4.2 MultiLayer Perceptron

MultiLayer Perceptron is a class of feedforward artificial neural network is another supervised machine learning model. MLP is sometimes strictly refers to networks composed of multiple layers of perceptrons. Perceptron is also a supervised learning algorithm capable of doing binary classifications according to a linear predictor function. The power of the MLP is the junction of several of those perceptrons that individually don't have the a good performance, but the combine performance can reach significant values.

This junction normally follows a standard architecture divided in layers, firstly a input layer, followed by one or more hidden layers and an output layer. The input and the output layers are the ones that interact with the exterior, being the first, as the name indicates teh one where the data should be injected at and the second where the result will be presented. The hidden layers is where the main learning occurs and where the most amount of fine-tunning is applied. In this entire architecture the number of layers and the number of neurons(singular cell inside a layer) can be tunning to improve the model performance.

4.3 Decision Tree

This supervised learning algorithm, as the name suggests, uses a tree like structure to support the data classification decision and aims to convert the observed patterns in the train data into conclusions about the data classes. The construction of this auxiliary structure is quite simple and straightforward, consisting in assigning a condition to each new branch, causing that the structure will converge in having only one class in leaf nodes, defining the conditions required to identify the existing classes. These condition choices must be taken with caution since a poor decision in this phase may cause a drastic decrease in the performance of the overall model. Many

times this process uses *Entropy* as a guide, where *Entropy* indicates, in a very high level approximation, the level of confusion/chaos/randomness of a set of information, thereby if the data tested belongs to only one class, the entropy should be 0 and the system knows that that data portion doesn't need to be divided anymore.

4.4 Random Forest

Similarly to the last model, *Random Forest* is also a supervised learning model and also uses tree structures to support the models' classification decisions. The difference is that in this approach the model uses several instances of Decision Trees, with the purpose of since sometimes Decision Tree don't generate the best performance possible, using multiple and dividing the problem so that not every tree specializes in all the features, dividing and conquering the problem.

4.5 K Nearest Neighbors

The last model that we decided to test was the *K Nearest Neighbors*, which although being one of the simplest learning algorithms, still is essential in the Machine Learning area. Still in the supervised learning algorithm area, this model uses a very simple approach to it's classification process; by representing every train data entry in a multidimensional space, once presented with new data, the model simple represents that new entry in the mentioned multidimensional space and verifies the class of the K nearest points(train entries), and the most frequent class in that K dataset is the one that is assigned to that new data entry.

5 Models Fine Tuning

5.1 Performance Evaluation

Once the classification models were implemented, we needed to measure their performance. This was accomplished through the confusion matrix, a matrix that correlates a model's predictions with the true values and gives us the number of: true positives (TP) and true negatives (TN), the correct predictions of wether an entry belongs to 1 of the classes or not; false positives (FP) and false negatives (FN), the incorrect predictions. The class defined as true can be any of the two, as the process is equivalent when inversed. In our case this was something done internally to the tools used, beyond our reach.

With these values, we were able to compute 4 widely used performance metrics:
- Accuracy, given by $A = \frac{TP+TN}{TP+TN+FP+FN}$
- Precision, given by $P = \frac{TP}{TP+FP}$
- Recall, given by $R = \frac{TP}{TP+FN}$
- F1-Score, given by $F = \frac{2 \times P \times R}{P+R}$

During our analysis we only consider *A* and *F*, as accuracy alone is not enough to evaluate the models (not robust to several aspects) and F1 correlates precision and recall very effectively.

5.2 Parameters Variation

In this section we present the manipulation of the hyperparameters related to each of the implemented algorithms, in search for better performances.

For the SVM, using the default parameters with the Radial Basis Function (RBF) kernel, the accuracy was around 55%. The sigmoid kernel proved to be more appropriate for the dataset as it increased both the accuracy to 57% and the F1-Score from 68% to about 73%. The remaining parameters such as the kernel coefficient (altered between "auto" and "scale"), tolerance for stopping criterion (ranged from 1.0×10^{-3} to 0.1), decision function shape or maximum number of iterations proved to have no significant effect when carefully altered.

Moving on to MLP, the default parameters of the neural network were the following: "relu" activation layer, "adam" solver for weight optimization, L2 penalty and tolerance both at 1.0×10^{-4}, "constant" learning rate and an architecturewith 3 hidden layers of 13 nodes. This configuration resulted in an accuracy of 54%. The best found configuration turned out to be the following: "identity" activation layer (although not very different from the original), "lbfgs" solver (that proved to be more effective than

"adam" for our dataset), a simples architecture with only 2 hidden layers of 10 and 5 nodes (that had the same effect as the more complex one), and the remaining parameters with the default values (as changing them only reduced at least one of the performance metrics). The final performance was of 56% of accuracy and 67% of F1-Score.

The DTs showed worse results in all metrics. Changing the criterion (function to measure the quality of a split) from "gini" to "entropy" showed no improvements, nor did tweaking the trees' maximum depth or minimum number of samples required to split an internal node. Defining the weights associated with classes made precision fall 2 percentiles. The best accuracy found was 53% with F1 at 58%. The addition of Random Forests was an attempt to improve these values, as RFs use several DTs to return a combined answer theoretically better than one considering only one DT. However, from our studies, Random Forests were only able to improve F1-Score, and only slightly with a maximum value of 60%.

Finally we dedicated our attention to KNN. Reducing the number of K nearest neighbors from 5 to 3 seemed to have a considerably good effect on its performance, raising 2% of the model's accuracy. Also, defining the algorithm used for computing the nearest neighbors was futile, as all alternatives had no effect whatsoever. Altering the distance calculation formula from euclidean to any other reduced the precision slightly. The best configuration gave an accuracy of 53%, with F1 at 60%.

5.3 Feature Manipulation

Lorem ipsum ...

6 Results Discussion

After executing all the parameter variation study, the feature analysis as well as some feature addition in hopes of adding some extra meaningful information to the already existing data, the best performance results were the ones presented in Section 5. Although not the most satisfying results, since when conducting a machine learning problem the aim is to obtain a model with a defined set of hyperparameters that generate a good performance with good values(close to 100%) on its performance indicators, but the results obtained in this study have a reason to be.

Referring to Section 3, one of the conclusions taken from analyzing the Figure 1 was that no obvious division was observable in the data projections in the different feature pair plane. Although this conclusion not absolute proof that the data isn't separable, the results of the tested model indicate that that is the case. This can be caused by several reasons, ranging from the dataset quality, the models used, the tunning of the hyperparameters or even the case of overfitting of the training data.

Through a deeper analysis over the data itself we didn't found any aspect that could compromise our study; the origin of the data and the metrics gathered seemed to be the most adequate for this type of study; the features also seemed to be widespread enough to encapsule all the possible signals related to this study, such as fatigue, immersion, stress, ... In relation to the models used, as already mentioned in Section 4 are state-of-the-art, therefor should perform well enough with the dataset presented, and for that matter, with any dataset. The parameters variation was probably the portion of the report study that took the longer, since we made a meticulous study of the parameters for all the models trained as detailed in Section 5. For that reason we can conclude that the parameter choice is also not the problem. Finally, in relation to the overfitting problem, the fact that all the models used gave similar results, it's quite improbable that all of them suffer from the overfitting. That allows us to also remove this problem from our list of possible issues.

Knowing that the variables in our reach are properly tested and in theory are correct, we can only conclude that the data itself is inseparable. This means that the two classes presented, the students from *DEI* and the students from *ESEC*, although obviously two different groups in this study, the conducted experiment indicates that in terms of the *RAVEN* intelligence estimation test there is no difference between the two groups.

Finally, in spite of all models produced a similar performance, where the *Accuracy* averaged in the 55% mark and the *F-Score* averaged in the 65% mark. This difference is due to the fact that both *Precision* and *Recall* are slightly higher than the *Accuracy*, which indicates that however the classification is almost random, in reality the models are able to generate more true results than negative results, which is quite interesting given the limitations observed from the models.

7 Conclusions & Future Work

Determining differences between populations is not a trivial task. The subjectiveness element of intelligence measurements increase the complexity of such problem, and so clear conclusions are very hard to find. Nevertheless there are many cases where, when the correct analysis tools are applied, solid conclusions are reached - even if they are precisely that no conclusion can be drawn - and this is exactly that case.

By discussing our results and the similarity of outcomes between very different classification models, and comparing our results with those obtained in the original study conducted by the dataset authors, we were able to understand that in fact there are no significant differences between Multimedia and Informatics students in terms of intelligence. In fact, this is parallel to our initial suppositions, as both classes require highly cognitive skills and problem solving capacities, even though applied to very different situations.

For future work, we believe it would not be a good idea to attempt implementing other classification algorithms, as they would very much likely return similar results. Instead, if the interest of analyzing the same dataset remained, we would probably address different questions such as "Are there any differences between women and men within these courses?", or "Is it possible to predict whether a student will get the answer right to one of the Raven's tasks or not?".

Our overall perspective on the project is very positive, even though we, as Informatics students, were wishing to find advantages on the side of the students with a similar background as ourselves. Nevertheless, we found the use of Jupyter Notebooks very practical and adequate to assignments such as this and, as it allowed us to segment the work to be done, we were able to keep a good balance and work distribution between group elements throughout the development.

References

[1] Warren B. Bilker et al., "Development of Abbreviated Nine-item Forms of the Raven's Standard Progressive Matrices Test", `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4410094`, accessed in December 2019.

[2] Felisa M. Córdova et al., "Identifying Problem Solving Strategies for Learning Styles in Engineering Students Subjected to Intelligence Test and EEG Monitoring", `https://www.sciencedirect.com/science/article/pii/S1877050915014787`, Procedia Computer Science 55 (2015), accessed in December 2019.

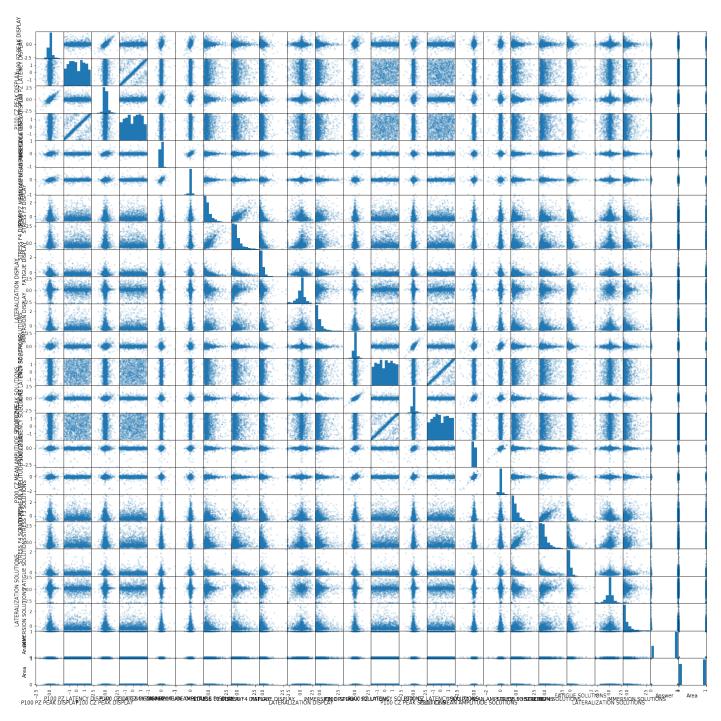[3] A. Tomé, "Data Mining Assignment", `https://elearning.ua.pt/pluginfile.php/1496406/mod_resource/content/3/ED_HCT_Raven.pdf`, accessed in December 2019.

Figure 1: Feature Correlations