Artificial Intelligence Project Report
The role of demographics in online learning
A decision tree based approach

Edoardo Canti

June 2022

Abstract

Trying to reproduce results obtained by Rizvi, Rienties, Khoja in section 4.1 of their paper. For this project we used Python programming language and ScikitLearn instead of R programming language and rpart package (like Rizvi et al.)

1 Introduction

Decision Trees are used in *supervised learning*. As every learning agent based on supervised learning it "looks" at some *examples* $(\overline{X}, y)$ where $\overline{X}$ is the input and y is the output.
Input and output are related by an unknown function f so that $y = f(\overline{X})$
In supervised learning the collection of such examples is known as *training set*.
Now the purpose of the learning agent is to find a function that approximate f, to measure the goodness of this function we use a *test set*.
Like every tree a *decision tree* is composed by *nodes* and *edges*. Every node correspond to a test on an attribute of $\overline{X}$ (a *feature*), and for each value in the domain of that attribute, starting from this node a branch represent the assignment of the current value to the selected attribute. The *children node* attached to the previous will now represent a test on another feature, knowing that the previous as already be assigned to a value.

2 The quality of a split

Decision trees algorithms uses a greedy approach to select the attribute (*feature*) on which apply the test at each level of the tree. The main idea is try to choose an attribute that reduces *impurity* as much as it can. A dataset is *pure* if contains all and only *examples attached to a class*. In each iteration the attribute selected to make the split is called *most important attribute*. In ScikitLearn to measure the quality (to find *most important attribute*) is used *Gini impurity* by default, and we still used it.

3 Implementation details

3.1 Dealing with the DataSet

While reading Rizvi et al. paper we understand how much it is important to pass the correct data to the predictive models (PredictiveTMA see below). For our purpose we need to work with students related to course A, registered in 2013, which attempted all five TMA's, which attempted final exam and never unregistered. It turns out that we had to deal with 289 learners (that would have been divided as 200 in training set and 89 in test set). *OULAD* dataset contains 7 tables. Two of these weren't important for our aim (vle and studentVle, representing the virtual enviroment and the student interaction with it). The other tables has been modified in order to maintain only those values useful to our purpose. To deal with this, we decided not to implement classes, but operate table by table and just implementing two functions: one used to connect studentsTMAs table with student info and the other used to encode features.

3.1.1 LabelEncoding

Is a function used to encode target values in numerical values in a range. If an attribute X has a not numerical domain Dom(X) with cardinality N a label encoding applied to X will transform Dom(X) into 0,...,N-1. This has been done because ScikitLearn implementation of decision trees doesn't support not numerical values in X set. However it supports not numerical in y set. To apply this operation we also came accross OneHotEncoding(). Here is why we didn't used it: For simplicity imagine now that region domain contains only three values: Ireland (I), London(L), Wales(W). Consider we are trying to convert region values into numerical using OneHotEncoding. What OneHotEncoding does is to create three distinct new dummy variables (and so attributes). Each object now will have these three new dummy variables indicating if the object itself has that variable true or false (0,1). So if now we will try to compute variable importance, we will not have a single variable importance for the variable region but instead we will compute the single importances of living in Ireland(I), London(L), Wales(S); this is not what we want.

3.2 PredictiveTMA

We decided to implement two classes.*PredictiveTMA* used to represent the predictive model of TMA's and final result, to which we attached all responsibilities of data handling such as splitting data in training set and test set; create the tree and run tree-related operations like compute predictions, return metrics like accuracy, recall, precision and collect variable importances.
So since we had to develop six predictive models, we used six instances of *PredictiveTMA* class to represent these models.

3.3 Tester

Since the aim was to collect results from all of the six models, the class *Tester* has been designed to handle the run of each single PredictiveTMA, and collect all metrics for each model. *Tester* also must be able to present to the user readable data such as: provide for each TMA a barplot representing the importance of each single variable, print a summary of the result showing metrics and confusion matrix for each model (Tester only shows the metrics, that are actually computed by PredictiveTMA), print a graph showing variables importances trend over time (over TMAs), return a DataFrame showing minimum, maximum and average value for each demographic characteristic (variable) over time (over TMAs).

4 Results

Rizvi et al. results showed that there is a link between demographics and learning outcomes. They also proved that this link changes during courses progression.
Our results also shows that demographic characteristics impact on student performances, some characteristics impact more than others and these impacts change over time. Turns out that the most important variables are:

- Region
- IMD band
- Education level
- Age band
- Gender
- Disability

You can see here some of the barplots that indicates for some TMAs how each feature impacted on the TMA score, and the plot showing the trend of variables importances over time (to see all of the plots please refer to the Jupyter Notebook).

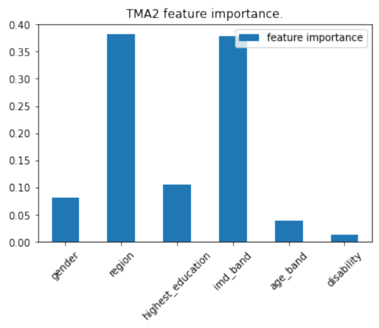

Figure 1: importances on tma2

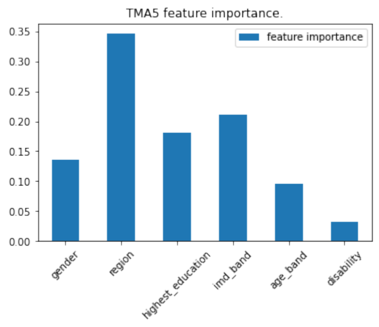

Figure 2: importances on tma5

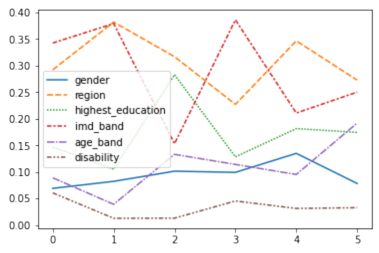

Figure 3: importances over time