Wroclaw University of Science and Technology

# Artificial Intelligence and Machine Learning



## Classifier Machine Learning Project

### Ines de Oliveira Soares, 256652

# Part I - Data Description

The database that I chose was listed on the UCI Machine Learning Repository website. It is called ‘Adult Data Set’ and the objective is to predict whether an income exceeds $50K per year based on census data. This dataset is also known as ‘Census Income’ and can be found on the following link: <http://archive.ics.uci.edu/ml/datasets/Adult>.

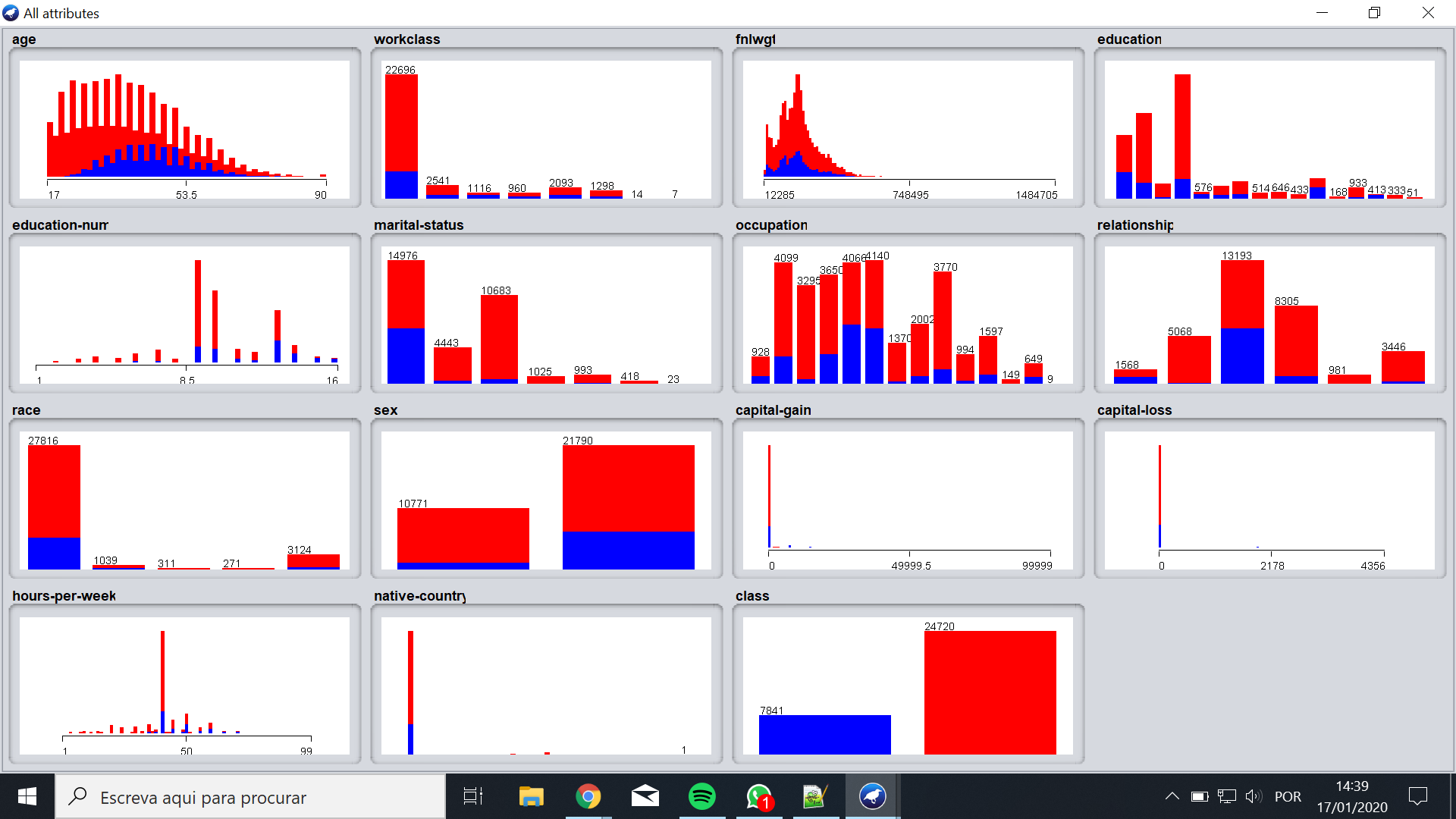
The ‘Census Income’ dataset is a multivariate dataset with 48842 instances and 14 attributes which are:

* Age: continuous
* Work Class: nominal
* Final Weight: continuous
* Education:nominal
* Education-num: continuous
* Marital-status: nominal
* Occupation: nominal
* Relationship: nominal
* Race: nominal
* Sex: nominal
* Capital-gain: continuous
* Capital-loss: continuous
* Hours-per-week: continuous
* Native-country: nominal
* Class: nominal (>50K, <=50K)

In this dataset 7% of the attributes have missing values, although because the number of instances and attributes is high, this shouldn’t be a problem.

# Part II - Preliminary analysis

To perform this task, I used the Weka software as suggested in the project class. In the picture below are represented the histograms correspondent to each attribute of the dataset.

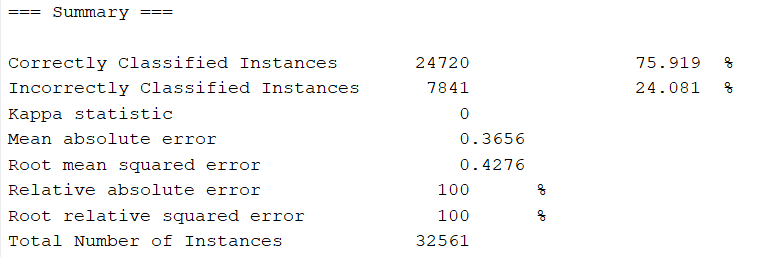


The class chosen for classification was the defalt one that evaluates if the incoming salary is greater than 50,000 dollars or not. This class has 2 unique values, therefore it has a reference level of ½. The discretization of the numerical classes was done automatically by the software. As we can observe in the histograms, each class behaves in a different form but it is notable that the class with higher income is inferior in all the class attributes.

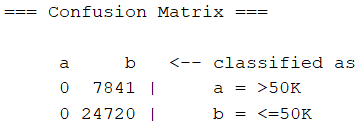
In order to verify if the accuracy was higher than the reference level, I used the Weka software with ZeroR and OneR to classify. If the accuracy was inferior than the reference level, we can conclude that it is impossible to automatically learn classification based on these attributes.

## ZeroR

With the ZeroR classifier, the accuracy obtained was 75.919% which is higher than 50%.

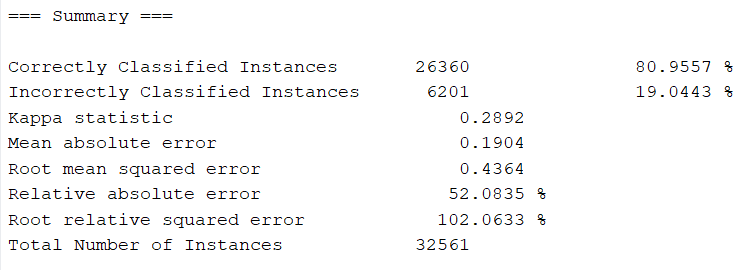


Although I verified (as you can check in the confusion matrix below) that the classifier evaluated all instances being less or equal to 50K. This method of classification in this case has a high value of accuracy because the majority of the instances belong to that class, otherwise the accuracy would be lower.

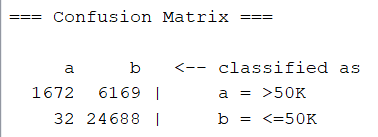


## OneR

With the OneR classifier the accuracy increased to 80.9557%.



In this case the classifier has a different criteria, therefore the confusion matrix is more credible.



# 

# Part III - Classification - the first experiment

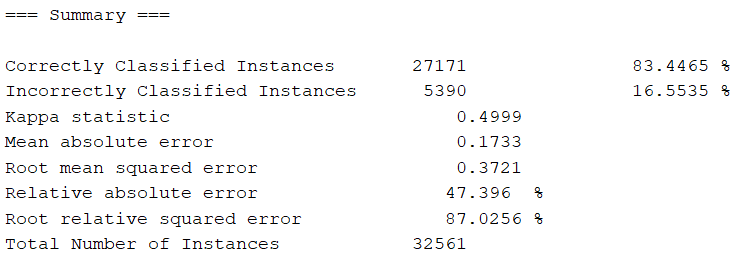
For this part of the assignment, I used the Naive Bayes Classifier for the first classification experiment, first using the training set and then using cross-validation, which is a first indicator of real learnability of the set. Then, in order to try to increase the accuracy, I used the Decision Table classifier both with training set and cross-validation.

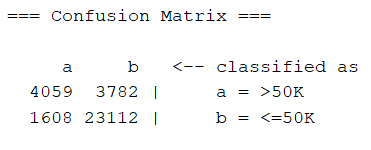
## Naive Bayes

The overall results were quite satisfying. The results presented below are definitely higher than the reference value and it was possible to increase by 3% the accuracy comparing to the previous classifier, reaching an accuracy of 83.4465%. The accuracy on the cross-validation is slightly lower (83.428%) which is expected.

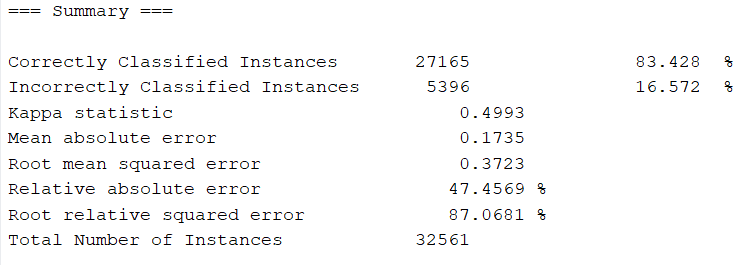
Below are also represented the confusion matrix for the training set and cross validation, where we can verify that the cross-validation compared to the training set has 6 more incorrectly classified instances in a total of 32561.

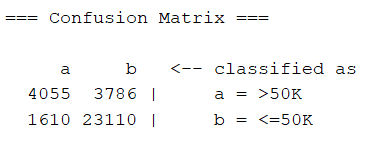
### Training set





### Cross-Validation

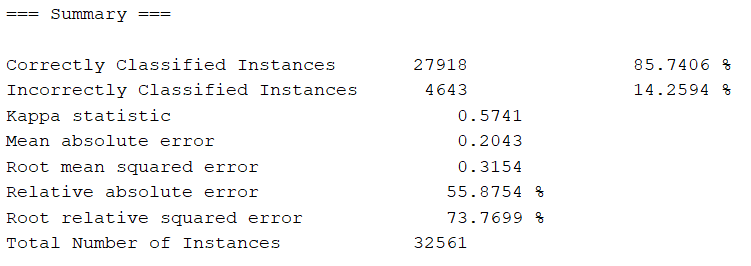


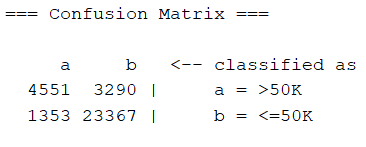


## Decision Table

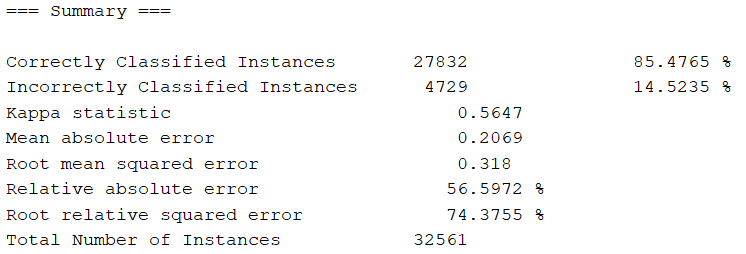
Using the decision table classifier I was able to increase a bit more the accuracy, this time to 85.7406% in training set and 85.4765% in cross-validation. The accuracy in cross validation keeps being lower than in training set which is expected as I explained before.

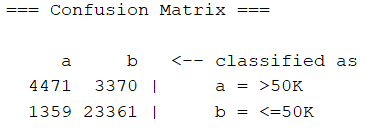
### Training Set





### Cross-Validation

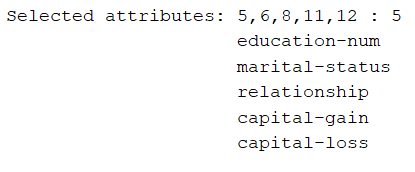




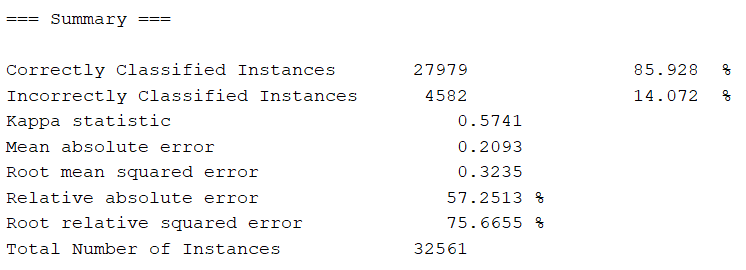
# Part IV - Minimizing error

## Selection of a subset of attributes

In order to select the most important attributes, I used the Evaluator CfsSubsetEval and the Search method GreedyStepwise and the obtained results were the following attributes: education-num, marital-status, relationship, capital-gain and capital-loss like it is shown in the picture below.

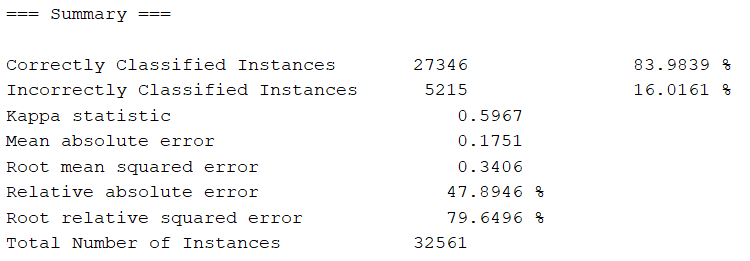


Like it was expected the accuracy increased a little bit more, this time to 85.928%. So the error decreased being now 14.072%.



## Discretization of numerical attributes

For the discretization of numerical attributes, in preprocessing I used the ‘Discretize first-last’ method with precision 6. But this time it was not possible to increase the accuracy, instead it decreased a little bit. Like we are able to see in the picture below, the accuracy value is 83.9839% and the error 16.0161%.

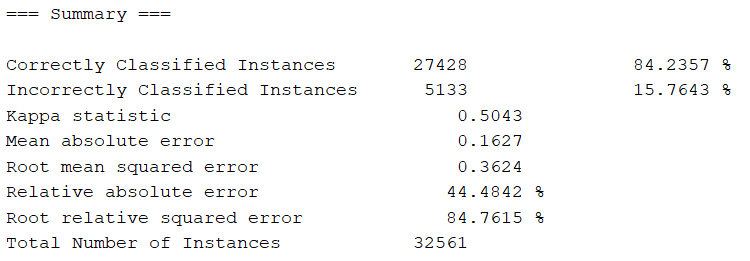


## Selecting a machine learning algorithm

Another possible method for decreasing the testing error is choosing a different machine learning algorithm. So I chose three different more: Multilayer Perceptron (or multilayer neural network), Random Tree and Random Forest.

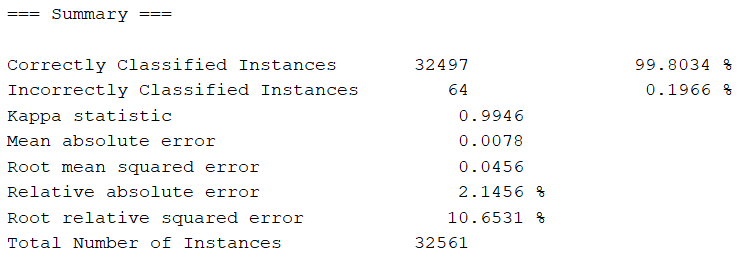
### Multilayer Perceptron

As it is possible to observe in the picture below, with the multilayer perceptron algorithm the error actually increases a little.



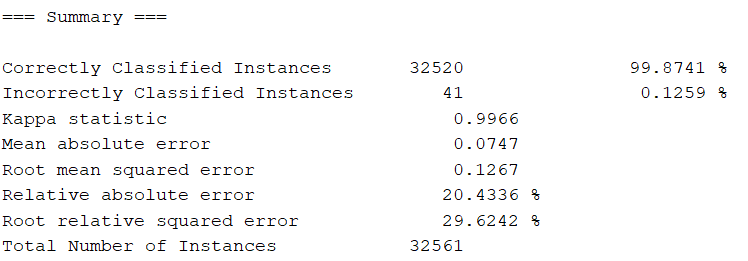
### Random Tree

On the other hand, with the random tree algorithm I obtained very satisfying results. The value of the accuracy increased to 99.8034% and, consequently, the error decreased to 0.1966%.



### Random Forest

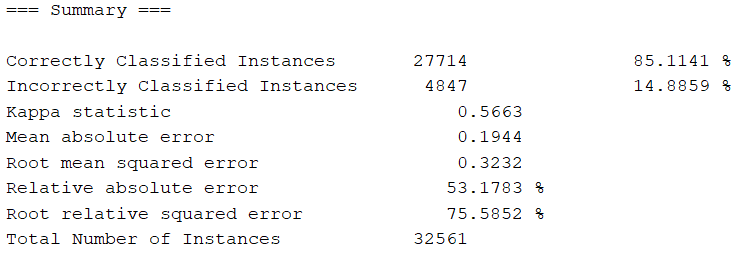
With the random forest algorithm the error decreases a little bit more making the results even better. Now the accuracy is 99.8741% and the error 0.1259%.



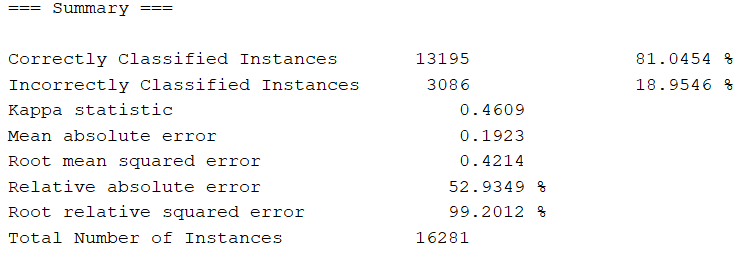
# Part V - Summary of the results

After all the error minimizing experiments, using the optimal training configuration, in this case the Random Forest algorithm using cross-validation, the obtained error on the test set was 18.9546%. Comparing to the error of the training set (14.8859%) is a little bit higher but still it is a reasonable value and can be considered to bring satisfactory results.

### Training Set



### Testing Set



## 

## Minimum number of samples

In order to determine the minimum number of samples required to train the classifier, I divided the dataset in various datasets with different number of samples, and the results are shown in the following table.

|  |  |  |  |
| --- | --- | --- | --- |
| # samples | error | # samples | error |
| 30107 | 19.0753% | 4839 | 20.8514% |
| 21030 | 19.5055% | 3489 | 21.1522% |
| 12033 | 19.4881 | 1219 | 21.0829% |
| 6415 | 19.9221% | 668 | 21.8563% |

Considering that the error for ZeroR method is 24%, an error of 21% in this dataset can be considered a bit high so I would say that the minimum number of samples is around 4000.