A logo for college computing

Description automatically generated

**Assessment Cover Page**

|  |  |
| --- | --- |
| *Student Full Name: Cristhian Elson Pereira Macedo* |  |
| *Student Number: 2024104* |  |
| *Module Title: Data Preparation* |  |
| *Assessment Title: CA1\_DataPrep\_HDip* |  |
| *Assessment Due Date: 05th April 2024 23:59* |  |
| *Date of Submission:* |  |

**Declaration**

By submitting this assessment, I confirm that I have read the CCT policy on academic misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source.

I declare it to be my own work and that all material from third parties has been appropriately referenced.

I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution.

Abstract

Contents

[Introduction 1](#_Toc158384946)

[Chapter 1 1](#_Toc158384947)

[Chapter 1.1 1](#_Toc158384948)

[Chapter 1.1.1. 1](#_Toc158384949)

[References 2](#_Toc158384950)

# Introduction

# First Steps

Imported the required libraries pandas and “numpy” to start the project, in case it needs other libraries it will be imported when necessary it comes. Loaded the dataset using the method “read\_csv” to import the required data existing in the .csv file.

The first command useful to start understanding the data is “.head()” to check the first five observations and features contained in the data frame after loaded. This command was important and necessary to get the first impression of it, returning that this dataset has 5 rows (features) and 37 columns (observations) thereon using the command “shape” shows us the complete body of the data in this case 4424 observations and 37 features to be analysed.

Use “.info()” next to Print a concise summary of a DataFrame. This method prints information about a DataFrame including the index dtype and columns, non-null values and memory usage. (The Pandas Development Team, 2022). Getting this information it was necessary to learn the names of features existing and the difference between the total of observations shown before in the “shape” command and their respective data types.

The command “.isnull().sum()” was essential in this part of inspecting the data set to know the existent missing values in each feature. With a more refined code “.isnull().sum()[df.isnull().sum() > 0]” it was possible to get just the specific features that exist missing values, in this case, the columns:

* Course;
* Father's qualification;
* Admission grade;
* Debtor;
* Tuition fees up to date;
* International;
* Curricular units 1st sem (grade);
* Curricular units 2nd sem (enrolled);
* Curricular units 2nd sem (grade).

# Data Cleaning

Before analysing the missing values, opted to check the types of data together with the file “student\_data\_dictionary.xlsx” provided, after some initial tests using “df.head()” noticed that wasn't possible to get the visual information about the 37 features and check their respective Type, so, in this case, opted to use the “.iloc” method, to get the first 13 features, after the middle 13 features and the last 13 features.

df.iloc[:, :13].head(5)

df.iloc[:, 13:26].head(5)

df.iloc[:, 26:37].head(5)

Analysing the information obtained using “.iloc” method and the command “dtypes” found different types shown in the imported file together with the data dictionary, from 37 total features, 12 features with different types, based on those described in the dictionary document and 25 features with their respective types correct.

**Result:**

12 features with different types, based on described in the dictionary document:

* **Marital status** is Categorical and should be Integer (int64);
* **Course** is Continuous and should be Integer (int64);
* **Father's qualification** is Continuous and should be Integer (int64);
* **Displaced** is Categorical and should be Integer (int64);
* **Debtor** is Continuous and should be Integer (int64);
* **Tuition fees up to date** is Categorical and should be Integer (int64);
* **Age at enrollment** is Categorical and should be Integer (int64);
* **International** is Continuous and should be Integer (int64) ;
* **Curricular units 1st sem (credited)** is Categorical and should be Integer (int64);
* **Curricular units 1st sem (grade)** is Continuous and should be Integer (int64);
* **Curricular units 2nd sem (enrolled)** is Continuous and should be Integer (int64);
* **Curricular units 2nd sem (grade)** is Continuous and should be Integer (int64).

25 features with their respective types correct:

* Application mode Integer;
* Application order Integer;
* Daytime/evening attendance Integer;
* Previous qualification Integer;
* Previous qualification (grade) Continuous;
* Nacionality Integer;
* Mother's qualification Integer;
* Mother's occupation Integer;
* Father's occupation Integer;
* Admission grade Continuous;
* Educational special needs Integer;
* Gender Integer;
* Scholarship holder Integer;
* Curricular units 1st sem (enrolled) Integer;
* Curricular units 1st sem (evaluations) Integer;
* Curricular units 1st sem (approved) Integer;
* Curricular units 1st sem (without evaluations) Integer;
* Curricular units 2nd sem (credited) Integer;
* Curricular units 2nd sem (evaluations) Integer;
* Curricular units 2nd sem (approved) Integer;
* Curricular units 2nd sem (without evaluations) Integer;
* Unemployment rate Continuous;
* Inflation rate Continuous;
* GDP Continuous;
* Target Categorical;

Before dealing with the different types, opt to understand and resolve the missing values to get the data for their respective correct types, whether this approach does not change some features it will be necessary to convert the whole feature to the correct type.

After using the “.unique()” method to check the missing values in the features that were missing values and different types, and just to make sure all features existent in this file, such as described below:

Marital status, Application mode, Application order, Course, Daytime/evening attendance, Previous qualification, Previous qualification (grade), Nacionality, Mother's qualification, Father's qualification, Mother's occupation, Father's occupation, Admission grade, Displaced, Educational special needs, Debtor, Tuition fees up to date, Gender, Scholarship holder, Age at enrollment, International, Curricular units 1st sem (credited), Curricular units 1st sem (enrolled), Curricular units 1st sem (evaluations), Curricular units 1st sem (approved), Curricular units 1st sem (grade), Curricular units 1st sem (without evaluations), Curricular units 2nd sem (credited), Curricular units 2nd sem (enrolled), Curricular units 2nd sem (evaluations), Curricular units 2nd sem (approved), Curricular units 2nd sem (grade), Curricular units 2nd sem (without evaluations), Unemployment rate, Inflation rate, GDP, Target.

It was found the different types of existents in the features below:

* Marital status\*\* = '?'
* Course\*\*\* = nan
* Father's qualification\*\*\* = nan
* Displaced\*\* = '?'
* Debtor\*\*\* = nan
* Tuition fees up to date\*\*\* = nan and '?'
* Age at enrollment\*\* = 'UnKnown'
* International\*\*\* = nan
* Curricular units 1st sem (credited)\*\* = 'Na'
* Curricular units 1st sem (grade)\*\*\* = nan
* Curricular units 2nd sem (enrolled)\*\*\* = nan
* Curricular units 2nd sem (grade)\*\*\* = nan
* Admission grade\* = nan

\* Missing Values

\*\* Different Data Types

\*\*\* Missing Values and Different Data Types

## Mark Invalid values as missing (NaN)

All different missing values existents such as before: “?, UnKnown and Na” have been marked now as NaN to get easy in case needed to catch them. Now that all missing values as marked with NaN, the “.unique()” method was used to check the missing values and make sure whether they have been changed or not after it was created a new data frame variable using the method “.dropna()” with the parameter “axis = 0” to drop the observations.

Using the methods “.isnull()”, “.values” and “any()” was checked the result: False that there were no missing values in the dataset next using “.dtypes” methods and checked if the respective types it was correct now, and not did any difference, the approach thought before to resolve the missing values after converting the whole feature to the correct type it was unsuccessful. There are no more missing values at the moment although the data types remain the same, so instead to deal with the missing values first it will try first to fix the data types.

After reading the Pandas Conversion documentation and trying to use “DataFrame.astype(dtype[, copy, errors])” Cast a pandas object to a specified dtype., “DataFrame.convert\_dtypes([infer\_objects, ...])” Convert columns to the best possible dtypes using dtypes supporting pd.NA., (The Pandas Development Team, 2024) and read the error “IntCastingNaNError: Cannot convert non-finite values (NA or inf) to integer” informed by the Jypter notebook., get understood that needs to deal with missing values first and next move to the others steps.

## Dealing with missing data

There are 13 features missing data: Marital status, Course, Father's qualification, Admission grade, Displaced, Debtor, Tuition fees up to date, Age at enrollment, International, Curricular units 1st sem (credited), Curricular units 1st sem (grade), Curricular units 2nd sem (enrolled) and Curricular units 2nd sem (grade). Let’s check one by one:

* **Marital status** is Categorical and should be Integer, there are just finite specific values to be part of the data such as those available on the data dictionary document, for this specific case, opted to drop the whole observation;
* **Course** is Continuous and should be Integer, same as Marital status, there are just finite specific values to be part of the data;
* **Father's qualification** is Continuous and should be Integer, same as the two before;
* **Admission grade** is missing a value but is Continuous, it is already at the right data type, opted to use the method “.mean()” to replace the NaN(s) value missing;
* **Displaced** is Categorical and should be Integer, limited values, same as before;
* **Debtor** is Continuous and should be Integer, limited values, same as before;
* **Tuition fees up to date** is Categorical and should be Integer, limited values, same as before;
* **Age at enrollment** is Categorical and should be Integer, with limited values, same as before, there is no way to guess which is the age;
* **International** is Continuous and should be Integer, limited values, same as before;
* **Curricular units 1st sem (credited)** is Categorical and should be Integer, limited values, same as before, in this case, I could use the method “.mean()”, but there are lots of observations with 0 as value, so opted to drop since data lost is minimal;
* **Curricular units 1st sem (grade)** is Continuous and should be Integer (int64), opted to use the method “.mean()” to replace the NaN(s) value missing;
* **Curricular units 2nd sem (enrolled)** is Continuous and should be Integer, dropped after drop **Displaced**;
* **Curricular units 2nd sem (grade)** is Continuous and should be Integer, same as “**Curricular units 1st sem (grade)**”;

## Dealing with data types

Change the data types for the features: Marital status, Course, Father’s qualification, Displaced, Debtor, Tuition fees up to date, Age at enrollment, International, Curricular units 1st sem (credited), Curricular units 1st sem (grade), Curricular units 2nd sem (enrolled), Curricular units 2nd sem (grade)]] = df[[Marital status, Course, Father\s qualification, Displaced, Debtor, Tuition fees up to date, Age at enrollment, International, Curricular units 1st sem (credited), Curricular units 1st sem (grade), Curricular units 2nd sem (enrolled) and Curricular units 2nd sem (grade).

# References

The Pandas Development Team (2022). *pandas.DataFrame.info — pandas 1.4.1 documentation*. [online] pandas.pydata.org. Available at: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.info.html [Accessed 16 Mar. 2024].

The Pandas Development Team (2024). *DataFrame - Pandas 2.2.1 Documentation*. [online] pandas.pydata.org. Available at: https://pandas.pydata.org/docs/reference/frame.html#conversion [Accessed 19 Mar. 2024].