

INDEX

[1. Introduction. 3](#_Toc77192285)

[2. Developing. 3](#_Toc77192287)

[2.1 Sequential time measurements. 3](#_Toc77192289)

[2.2 DoParrallel. 4](#_Toc77192290)

[2.3 Lapply. 5](#_Toc77192291)

[3. Comparative. 6](#_Toc77192292)

# 1. Introduction.

# The project that we can see in this document consists of the parallelization of a code provided by the course mentor (Janez Povh). The aforementioned code is made under the programming language called "R" and allows us to make a prediction using various statistical methods.

# 2. Developing.

## In this section, we will see how the code has been parallelized using two methods: "Doparallel" and "Lapply". In the same way, we will see how we have modified both the sequential code and the parallel code to be able to measure the execution times of each section of interest of the code, as well as the total execution time.

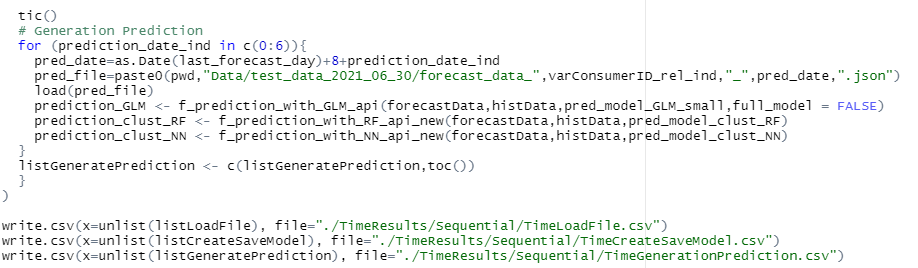
## 2.1 Sequential time measurements.

For this section, the code provided by the mentor has been understood and has been modified following the steps that we can see below:

* We have added the tictoc library.
* Inside the main loop we have added the function “System.time” (it allows us to be able to measure the total execution time).
* The tictoc library has been used to measure each of the parts of interest (load files, create and save the model and generate the prediction).
* We save each calculated time in a separate Excel file.

Next, we will see the modified code section.





## DoParrallel.

For this section, we have modified the code obtained from the previous section under the following steps:

* A function called "ExecuteCode" has been created that, through a parameter that we send to it, allows us to be able to carry out all the main code for a single consumer.



* An external loop has been created that allows us to go through the list of consumers using a “Foreach” and the number of cores to be used has been registered.

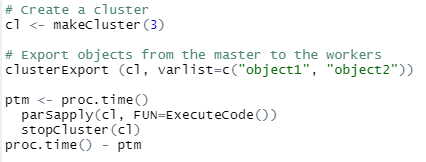
Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

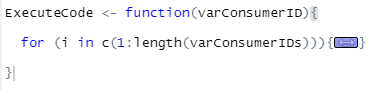
## 2.3 Lapply.

For this section, the changes that we can see below have been made:

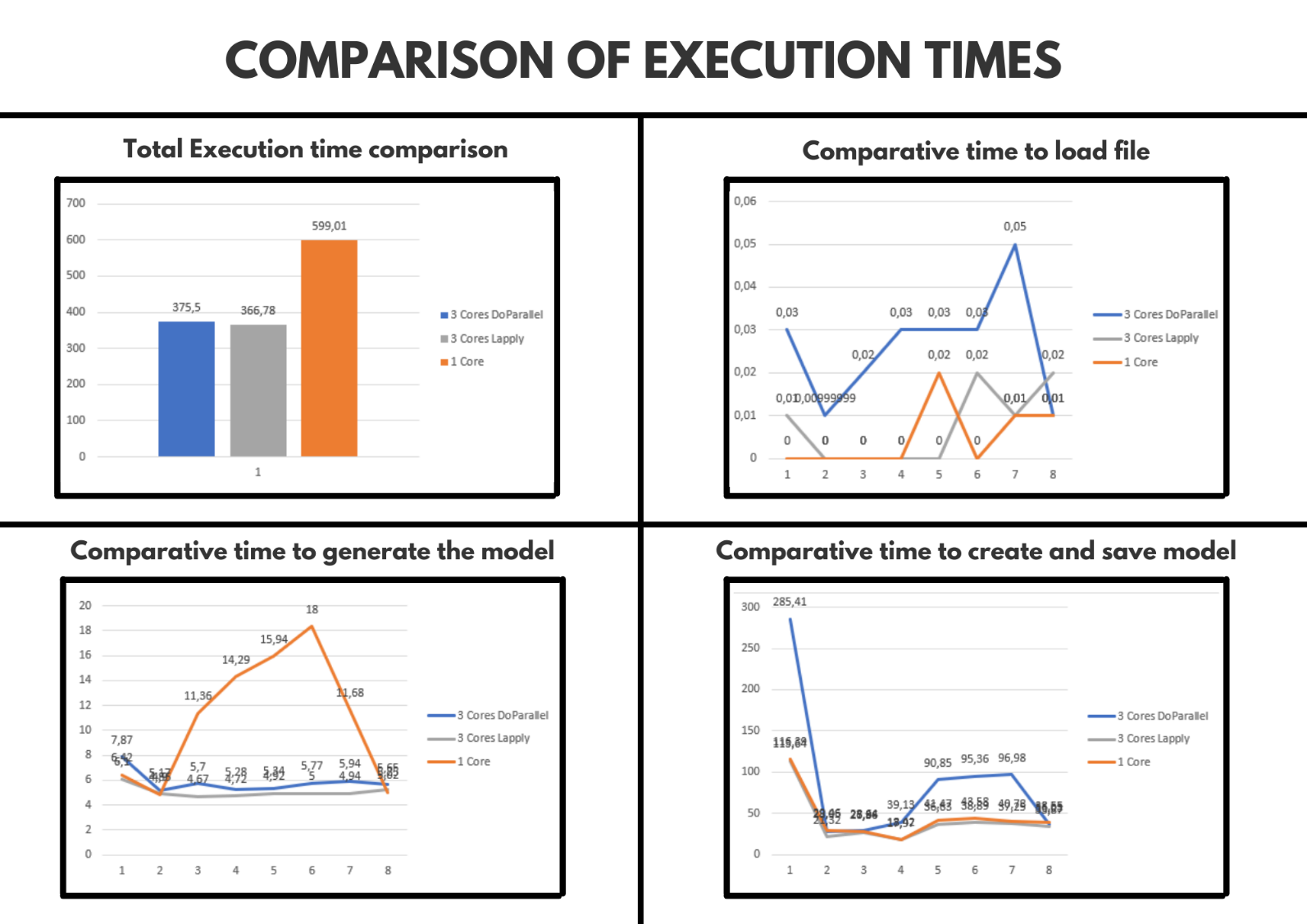
* A cluster with 3 cores has been created and added to the “parSapply” function together with the name of the function to be executed.



* The loop has been added inside the function to gain more simplicity in the code.



# 3. Comparative.

Next, we can see a comparison of the execution times obtained for each of the implementations made.

As we can see in the previous image, the highest execution time obtained is in the sequential version. Similarly, it should be noted that the times obtained in each parallel version are more or less similar, therefore, we should not opt for any at least for now.