IE LAW SCHOOL
MASTER IN LEGAL TECH
CODING AND PROGRAMMING I: TECHNOLOGY BASICS APP BUILDERS
PROFESSOR: BOGDAN-ALEXANDRU RAȚIU
STUDENT: EDUARDO MOTTA DE MORAES

PROGRAMMING AS A LAWYER: SUMMARIZATION OF LONG TEXTS

I. Background: anyone can learn how to code to solve everyday problems

Technology has been evolving more and more rapidly, with promising advances in fields such as artificial intelligence and cryptography. Naturally, while everyone must learn about and adapt to these changes as they are becoming more common and relevant, the development of cutting-edge innovation should arguably be left to the specialists in the field.

That does not mean, however, that other professionals cannot benefit from a hands-on approach when learning how technology works. In the legal sector, as in many others, there are a myriad of tasks that can be automated with no more than a basic knowledge of coding, especially nowadays with popular high-level language options like Python.

One such example is the analysis and summarization of text using natural language processing.

II. Problem: working in the legal sector requires a vast amount of reading

Lawyers have always been known for, among other things, spending long hours reading vast amounts of documents and studying case law. Every time a new case must be prepared for, there is the need to be up-to-date on the relevant subjects, which requires long periods sitting in front of a computer trying to absorb the content of

many pages of text. Most of the time, the majority of the material that is analysed is not entirely relevant to the case – but the professional will only know that after reading all of it, or at least skimming through it.

This is a prime example of a common boring task that nowadays can be partially automated with some knowledge of coding: instead of reading every single line of every single document, a lawyer can program a computer to do that and return only a summary containing the most relevant sentences. Potentially, this could save hours of work, freeing the professional to pursue more relevant activities.

But to understand how this is possible even without a deep knowledge of programming, one must understand the way programming languages work and the flexibility that they allow us.

III. Solution: how Python can be used to automate summarization

If asked what a programming language is, most people who do not work in the technology field will think of an unintelligible jumble of letters and symbols on a computer screen. But it is relevant to notice that, while some languages are indeed harder to read (low-level languages), others are closer to what a human language looks like (high-level languages) and are much easier to learn and understand.

Python is probably the most popular example of a high-level language today, having a simple syntax and yet allowing the development of powerful applications. Besides that, Python is known for having a vast collection of libraries, or modules, that contain code written by other, more experienced programmers – that can be easily implemented even by a beginner, once he learns the basics of the programming language.

One example is called Sumy, a module that generates the automatic summarization of texts using natural language processing techniques. Natural language processing is a subfield of linguistics and computer science, concerned with the analysis and extraction of relevant information from written documents. As noted, this is one example of a task that can be automated to save hours of work for lawyers.

There are many techniques available for the summarization of texts, some of which are available with Sumy: (i) Latent semantic analysis (LSA), an unsupervised method that combines term-frequency techniques with singular value decomposition; (ii) Lex Rank, a graphical-based summarizer; (iii) Luhn, one of the first algorithms, that scores sentences based on frequency of the most important words; and (iv) Text Rank, similar to Lex Rank but with different results.

As mentioned, the implementation is relatively simple. Using Python, it is just a matter of importing all the necessary Sumy modules and, with a little knowledge about how summarization works, choosing the appropriate technique. With a basic understanding of the programming language, it should be trivial for any lawyer to utilize this to simplify his or her work. It is worth noting, moreover, that the application may also be used for other purposes, such as summarizing news articles and academic papers.

IV. Implementation: analysing the code

Sumy can be used in the command line, which for some people may be enough. For this example, though, we chose to go a step further and deploy a graphical application using the module Dash and the hosting service Heroku, making for a more user-friendly experience. The code can be found here and the application here.

The first step is to import all the necessary modules. Afterwards, we must instantiate the Dash application, define its HTML layout, and define a function – in this

example called “update_summary” – that will do the summarization using Sumy. The application is updated automatically with the return of the function by using the Dash “callback” method as a decorator.

For the Dash application, we chose to have a dropdown menu for selecting the language of the text being summarized, as well as a text box for the number of sentences of the summary, and a way to choose the technique to be used. The input text is typed in a larger text box and the summary is provided below in the white area.

In the function update_summary, we must have: the parsers, to be used depending on the source of the text that is being summarized (HTML from a URL, or plain text); a tokenizer (to separate the text into parts that will be analyzed); the summarizer method (LSA, Lex Rank, Luhn, or Text Rank); a stemmer (to reduce words to its root before the analysis); and a list of stop words (words which will not be taken into account by the summarization algorithm).

Deploying the application to Heroku requires following a simple tutorial that can be found in the Dash website.

The end result is a web application that will provide the summary of the input, whether it is a URL or just pasted text. Since no knowledge of Python or the command line is necessary for using the application, it can also be shared with other people who might have the same need, but not the time or patience to learn how to code.