SAE S1.01
Project "Automatic classification"

TIVOLLIER Jarod
BUISSON Léane
D2

- An intro introducing the topic

The objective of the SAE 01 is to create an automatic learning method, enabling us to analyze and classify news. It must be as efficient and fast as possible, but it must also be reliable. For this, we use the java intelliJ IDEA software. It is on this programming software that we will gradually compose our method while analyzing the many faults and problems in order to be able to optimize our program. Different classes and files are available to us. The news file groups together all the news to be classified. These are put in order by 5 categories, environment and science, culture, economy, politics and sport. Each news has an associated number and contains a text of a few lines. To carry out this classification, we are going to create a lexicon allowing us to rely on the same lexical field. For example, for a news item in the Economics category, the words budget and finance are the links. In the same way for all the other categories, a set of “marker” words are associated with it and this is precisely the purpose of creating the lexicon. Moreover, each word is associated with a weight corresponding to the degree with which the word indicates membership in the category. The weights range from 1 to 3, depending on the relevance of the words. The max value of the weight, a maximum weight is of value 3, means that the word strongly defines the targeted category. Take the Sport category for exemple, that can be associated with the following words: sport, cup, match, player, home. If we associate them each with a weight we get: sport: 3, cup: 3, player: 2, home: 1. This is what we are going to deal with in the first part of the project, by creating lexicons from words chosen from among those contained in the news. Then in a second part these lexicons will be built automatically, in the way of machine learning in the field of artificial intelligence, that is to say to learn from a certain amount of data and have the ability to improve their performance at solving tasks without being totally programmed for each one.

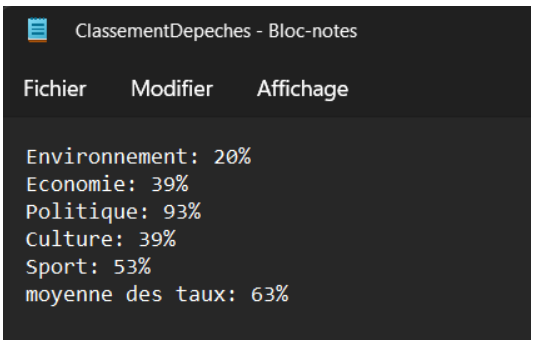
- Focus on what we did / did not do

Now let's look at all that we were able to achieve during the first part of the project. As said before, we first created lexicon files containing about twenty words to have an overview and a test of what it will give later. Next, we created 5 files, manually, Environment-sciences, Economy, Culture, Politics, Sport, each containing twenty words selected as being relevant to illustrate the category.
We started coding in Java with the creation of attributes and associated accessors and constructors. We then initialized the lexicon attribute in the method “InitLexicon”, which is a list of several words, from a random lexicon file which will then be modified according to the wanted lexicon. Thanks to the files previously created we were able to test our initialization of the lexicon, by displaying the loaded lexicons. Then we created “IntegerForString”, a search method used to check for the presence of a word, or string, in a list by going through it entirely. If what we are looking for is not present, we associate it as a value of zero, and vice versa we associate it with an

integer that corresponds to its character string, if it is present in the list. The test of this method is carried out with the entry of a word by the user, this word will be assigned to a weight in the lexicon previously loaded. The "score" method is a method that we coded, used to distribute a score according to the news. In this method, we call the previous method, to get the integer of the word, the string, back and add it up to get its score. We then developed a method “StringMax” to compare the associated integers, and thus find the string with the highest score and being the representative of the list. We then test this method with our five lexicon files per category, it then returns the name of the category with a maximum score. Just like the "IntegerForString" method, the method “IndexForString” will go through the entire list and return, not an integer, but a clue associated with the string. We have also created an averaging method. The average is calculated from the integers stored in the list, we sum all the integers and we divide by the number of integers added together. Then, “RankingNews”, this method creates a file containing various information, such as the percentage of categories with high-weight news and the average of these percentages. The "InitDico" method is useful to create a "dictionary" of a set of words contained in the list of dispatches, where it will appear at least once, as well as its associated score. To continue, “CalculScores”, updating the scores in the "dictionary" is done from this method. It makes a complete roam of the list while comparing if the word exists in the "dictionary", it will be linked to a score that will either be subtracted by 1 if the news is present in the active category, or will be increased by 1 if the news is not present in the active category. All the scores of each word contained in the lexicon are rectified to simplify comprehension, with the method “WeightForScore”. Because of this, words with a score of less than zero will have an insignificant value of -1, words with a score between zero and three will have a value of 1, words with a score between three and seven will have a value of 2 and finally words with a score greater than seven will have a value of 3. Thus, we have an order of importance between the different words, those that are of value 3 are representative of the category and those that are of -1 are insignificant. The final method that we created allows us to automatically generate lexicons, it puts all the previous methods together to create a new lexicon file for each category. These five new files will serve as lexicons instead of the old ones, which are more accurate and complete.

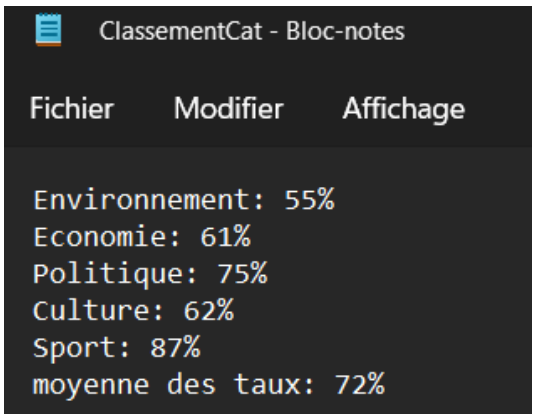
- Presentation of results, discussion of results

Among all the methods created previously, some were tools and others giving results that we will analyze below.
First of all, we obtained the ranking of the file gathering the news, with all the percentages of each category. According to the image opposite, in the manually generated lexicons, the Politics category holds the highest rate therefore linked to a fairly large number of scores. The results are less accurate due to the fact that they are derived from a manually generated lexicon, containing fewer words, thus creating a rate distorted by lack of information.



Secondly, a second result, the generation of the lexicon of the culture category. We can observe that the words come from all the news of the culture category, without taking into account the words present several times. They all have a number that therefore represents their weight, as explained earlier, we can see here that the word "théâtre" has a value of 3 which means that this word is unique to the culture category and that it is therefore representative of it. Conversely, the word "est" has a value of -1, because it is a term used in several categories and it is not a term that can define the culture category.

Finally, as a last result, we recreate a ranking but this time with the new lexicons generated. Here the results are more precise, since we are based on a larger database, because indeed there are many more words. This time, the category with the highest rate is Sport. This result being more correct, it is on this lexicon that we must base ourselves to make any study.



- Complexity analysis of methods

Now let's move on to the analysis of complexity, which is a crucial aspect to address in order for the program to be designed in the best possible way.
Despite the fact that the “Score”, “News” and “CalculScore” methods are optimized thanks to the tool method call, many of them use a lot of loops and a lot of conditions. This engenders a large number of comparisons in the program, the algorithmic cost is greatly increased. For each news and for each category, a number of comparisons equivalent to 500 for news and 5 for categories must be made, which means that a large number of results to be compared must be analyzed.

- Conclusion (prospect for improvement)

The best way to optimize our program would be to reduce all these comparisons by limiting the number of loops and creating many more tool methods. This project will have allowed us to open up in a new type of programming with the machine learning system, or the beginning of artificial intelligence. Unfortunately, we lack many notions to improve this program which could be shorter, more optimized in time and

especially in algorithmic cost. The concept of automatic learning being new, we were able to learn many things, in particular the fact that automatic learning is composed of two phases. The first state is a phase of implementation of the programs, where we created methods serving as models from data. This state is also called a training state, which simply boils down to the use of the model provided, the machine stupidly applies what we have previously initialized and brought to it. These models will then be used again in the second state, where the complete program uses each of these methods to adapt them. The second phase is therefore used for adaptation and production. After we have defined the model, the predefined data can be replaced by others to obtain a result that suits the user's request. The system we have could be improved by providing the algorithms with more training data. Thus it has more data and it therefore becomes effective in responding to the needs of the user.