Exercise 1: Extracting entities from unstructured data

The aim of the exercise is to propose a solution to extract information for an unstructured document like a pdf. Here we used Swiss Re Financial Report (a pdf document) to extract and report all the locations mentioned and the pages where they where found.

First we had to list all possible locations that could be found in the document. For the countries we used a python package named “pycountry” that lists all countries in the world. For the cities, it was a bit more complicated as listed all cities in the world gives to much possibilities leading to a time consuming program. That is why, we went on the company website Swiss Re to find all the office locations and we listed them by hand in a txt file. We also checked the pages with a lot of locations (e.g., page 282) to add the missing locations.

Second we used python (“pdfminer” package) to load the pdf document and to check for each location listed if they appear in the document and where (page number). It was useful to print in txt files each pages load by the python program to understand how the text is converted. In fact some locations appeared in upper cases in the pdf documents and where converted in lower cases in the python program. That is why we had to convert all the documents and the locations’ file in upper cases to prevent mismatches (i.e., the program does not find a location in lower cases from the locations’ file, if it is in upper cases in the pdf document). A problem was also the non-alphanumeric characters before and after the locations. It was a problem for example for the Carouge location that was delimited by a special character. To find all locations despites of these characters we looked for /W”location”/W. The /W is used to delimited the locations by non-alphanumeric characters. Once the location file ready and the pdf document loaded we found the locations in the document with the python regular expression package.

Finally we printed the found locations with their associated pages in a result document.

The accuracy of the solution provided is good. However it would be better to directly have the location file with all the possible locations we have to find in order not to forget locations.

For the generalization of the program, we have to change the input file. The first challenge is that to get a quicker program for the Swiss Re document we only listed the cities where Swiss Re has offices. For other documents other cities might be cited. That is why to generalize the program, it is important to list all the cities in the world even if the program will be much more time consuming to be sure to extract all the location of every document. Second, the written program is working for pdf documents. For other types of documents (e.g., csv, txt, xml) we have to load them with other python packages (e.g., csv, xml.etree.ElementTree).

Exercise 2: Classifying news into categories

The aim of the exercise is to propose and implement and efficient classifying method in order to classify a set of news articles from the web in several categories like Business, Politics, Technology, Sport or Entertainment.

Building such a classifier with defined categories needs supervised classification methods. In order to implement a supervised classification method, a training dataset is required. The training dataset allows to the classifier to learn the relevant information about the features that defined the different categories. About the classifier, a lot of different algorithms are implemented like the Naive Bayes classifier, the maximum Entropy Classification and the support vector machines.
For the purpose of classifying news articles in six different categories that are well different, using a simple and efficient classification algorithm like the Naive Bayes classifier might be reliable. The Naive Bayes classifier use a probabilistic approach that assumes that all the features of the training examples are independent of each other given the category. For the news articles classification we can make this assumption.

We measure the classifier performance with a testing dataset by applying the classifier on the testing dataset and comparing the classification made by the classifier with the known classification of the testing dataset. A high similarity between the classifier and the known classifications means that the classifier is accurate and relevant. It is important to choose randomly the training and the testing datasets and to process the classifier several times in order to measure the classifier performance with more reliability.

The first step in the classification process is to load and to prepare the data for the classification algorithm. The data preparation is very important, as it will condition the features of each news articles, which are essential for the Naive Bayes classifier rules. As we compare the words of the news articles, we have to tokenise and to lemmatise as well as to standardize them in lowercase in order to compare the frequency of occurrence of each word according to their news article category. Then it is essential to take a large enough training dataset that the classifier identifies accurate specificity of each category. When the classifier gets the features of the categories, it is able to classify the testing dataset. The testing dataset really allows evaluating the accuracy of the chosen classifier. After an evaluation of the classifier showing its good performance, we can use the classifier to classify new news articles according to the defined categories.

Applying the classifying method explained above on the BBC articles showed that the Naive Bayes classifier is efficient for classifying news articles. Taking 70% of the BBC articles as training dataset allows the classifier to classify in the good categories around 90% of the testing news articles. Implementing two feature methods, one counting the words occurrences and the other one just detecting

them in the news articles, can be interesting to compare the efficiency of the classification. In our case the feature methods gave similar results. As the Naive Bayes classifier gave concluding results it was not useful to implement another (more complex) classifier.
To conclude, implementing a classifier needs a large enough dataset to train and test it. It is also important to clean the data and to choose an efficient feature rule as well as a good classifier.

Exercise 3: General Data Science questions

The aim of the exercise is to explain a method to detect particular patterns in very large pdf documents. As we have to analyze big data, we will use big data methods as Hadoop.

Hadoop is a framework that can process a large amount of documents by dispatching them to cluster of computers with two programs: Hadoop distributed file system (HDFS) and MapReduce. Using this framework will save time.

In order to find similar patterns between the documents we have to implement an algorithm with different steps. The first step is to characterize each document with a set of short strings. Then it is possible to visualize each document in a matrix by representing their set. The second step is to compare each set together by calculating cosine similarity between given two sets.

Hadoop and the describe algorithm are adapted to big data, that is why the duration of the program should not be too long.