COMP 6721 Applied Artificial Intelligence (Fall 2021)

Lab Exercise #11: NLP, VSM & NER

Question 1 Our first exercise is to convert natural language text into a vector representation that we can later use for further processing, for example, as input to a machine learning algorithm.

(a) Write a Python program that computes the binary sentence vectors for the three sentences on Worksheet #10 (exercise "Vector dot product").

(b) Now compute the dot product as asked on the worksheet.

(c) What is the meaning of the dot product in case of binary vectors?

Question 2 Instead of binary vectors, we will now compute tf-idf vectors.

(a) In Lab #4, you already used scikit-learn's `CountVectorizer()`. Modify your program to compute tf-idf vectors using the `TfidfTransformer` on the created count vectors.[1] Print out the tf-idf vectors, as well as the df values for each word in the corpus. To validate the calculations, make sure you understand what settings are being used by default (see the documentation on the parameters regarding idf reweighting, smoothing, and sublinear tf scaling).

(b) Now compute the *cosine similarity* matrix between the documents and print it out.[2]

Question 3 Let's now try to use the Python spaCy NLP library[3] for *Named Entity Recognition* (NER). NER is the task of identifying key information/entities in texts. Here, entities can be a single token or span a series of tokens. Every detected entity is then classified into a predetermined category (e.g., *Person, Organization, Date*). NER is an important step for various NLP tasks, for example, when you want extract the main subjects or concepts from text passages or get an overall idea of what a text is talking about.

In spaCy, named entities are available through the `ents` property of a `doc` object. Try the following code to extract named entities and some of their properties:

[1] See https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html#sklearn.feature_extraction.text.TfidfTransformer

[2] See https://scikit-learn.org/stable/modules/metrics.html#cosine-similarity

[3] See https://spacy.io/

```
import spacy

nlp = spacy.load("en_core_web_sm")
doc = nlp("Apple is looking at buying U.K. startup for $1 billion")
for ent in doc.ents:
   print(ent.text, ent.start_char, ent.end_char, ent.label_)
```

(a) Try playing a little with different sentences and visualize what labels are trained to be predicted by spaCy.

(b) Use *displaCy* visualize the identified entities.

(c) Now try restricting your visualizer to only display specific categories of named entities for the following sentence. First try to visualize all the named entities to make sure you are only visualising the ones you wanted to see, e.g., only entities of type *Organization*:

```
text = "European authorities fined Google a record $5.1 billion on
Wednesday for abusing its power in the mobile phone market and ordered
the company to alter its practices."
```