CA – Assignment 1: Data Acquisition

- Group: `FakeNews`
- Group members:
  - Adnan Manzoor
  - Sajjad Pervaiz
  - Kevin Taylor
  - Christoph Schäfer

Structure

```
|-- data-acquisition-assignment
    |-- code
        |-- data-unification.py
        |-- preliminary-statistics.ipynb
        |-- statistics.py
    |-- data
        |-- ArgumentAnnotatedEssays-2.0
        |-- UKP-InsufficientArguments_v1.0
        |-- UKP-OpposingArgumentsInEssays_v1.0
        |-- train-test-split.csv
        |-- unified_data.json
    |-- documentation.pdf
    |-- README.md
    |-- requirements.txt
```

How to reproduce the unified data file

- Make sure you have the same file structure as above(otherwise set the paths at the beginning of `data-unification.py` and `preliminary-statistics.ipynb`)
- Install on a `venv` the packages listed in the `requirements.txt`
- Make sure that `en_core_web_sm` is available. Download it via `python -m spacy download en_core_web_sm`
- The unified data is saved in the file: `/data/unified_data.json`.
- To reproduce the file, run the `data-unification.py` in the `code` folder with the following command:

```
python data-unification.py
```

How to run the preliminary statistics:

The methods to calculate the preliminary statistics are located in the file `statistics.py`. If you want to run the file, use the following command:

```
python statistics.py
```

Otherwise we also added a jupyter notebook file `preliminary-statistics.ipynb`. The jupyter notebook can be opened by the following command

```
jupyter notebook preliminary-statistics.ipynb
```

Calculation of most specific words

> Most specific words for argument unit are those that appear more often in the argument unit but not in other units in the text (including non-argumentative units)

To calculate the `most specific words` for the three different argument units (major_claim, claim, and premise) , we used the `TF-IDF` score.
TF-IDF stands for `Term Frequency – Inverse Document Frequency` and is often used in information retrieval and text mining. The TF-IDF `score/weight` is a statistical measure used to evaluate how important a word is to a document in a collection or corpus.
The importance of a word increases proportionally to the number of times the word appears in the document but is offset by the frequency of the word in the corpus (TFIDF.com).
It also can be successfully used for `stop-words` filtering and other words that appear very often. In our case if a word appears very often in one argument unit but also in other argument unit(s). Therefore we calculate a score for each word based on its relevance and frequency for each of the argument units (major claims, claims, and premises).

TF-IDF calculation is defined by

```
TF-IDF := TF * IDF
```

where TF and IDF are defined as:

```
TF(t) := (number of times term t appears in a document) /
                    (Total number of terms in a document)
```

and

```
IDF(t) := log_e(Total number of documents /
                Number of documents with term t in it)
```

So in our application:

- `TF score` is calculated for each word in an argument unit. The text for each of the argument units from all the train-split essays is combined to form 3 argument unit texts (one for each major_claims, claims, and premises). Then the formula described above is applied on each word of the argument units.

- For `IDF`:

```
Total number of documents = total number of essays in train-split = 322
```

And

```
Number of documents with term t in it  = number of essay texts in train-
split with term t in it.
```

So for example if a word `w` appears very often in an `argument unit`, it leads to a high TF score but if it also appears very often in other argument/non-argument unit(s) the IDF score will be very low. Therefore, the `TF-IDF` score will be very low. Consequently, we get only a high `TF-IDF` score, if the frequency of the word `w` is very high in an argument unit and the word `w` is not used very often in other argument/non-argument units. Therefore, we have calculated the `most specific words` which are very frequent for each argument unit but less frequent in overall train-split corpus/essays.

Specific examples would be the following:

1. The word `students` appears as follows:

```
    major_claims = 45 times
    claims = 112 times
    premises = 236 times
    Number of essay texts with word 'students' in it = 85

    IDF score = ln(322/85) = 1.33

    TF score for major_claims = 45/8788 = .0051
    TF score for claims = 112/18139 = .0062
    TF score for premises = 236/53211 = .0044

    TF-IDF score for major_claims = .0051*1.33 = .0082
    TF-IDF score for claims = .0062*1.33 = .0068
    TF-IDF score for premises = .0044*1.33 = .0058
```

Since the word appears frequently in all argument units and also not that frequently in whole of the train-split corpus/essays, it gets a high TF-IDF score in all 3 argument units and is chosen as top-10 specific word for each.

2. The word `people` appears as follows:

```
major_claims = 79 times
claims = 208 times
premises = 435 times
Number of essay texts with word 'people' in it = 276

IDF score = ln(322/276) = 0.15

TF score for major_claims = 79/8788 = .009
TF score for claims = 208/18139 = .0115
TF score for premises = 435/53211 = .0082

TF-IDF score for major_claims = .009*0.15 = .0013
TF-IDF score for claims = .0115*0.15 = .0017
TF-IDF score for premises = .0082*0.15 = .0012
```

Even though it appears frequently in each argument unit, it also appears in so many essay texts and hence gets a really low IDF-score and hence is not considered in Top 10 of `most specific words`.