Assignment III: Argument Mining

### Installing Dependencies:

- Assuming you have a working ``Python`` library and a ``pip`` tool:
  * Install `jupyter` using the command: ``pip install jupyterlab``
  * Install `spacy` using the command: ``pip install spacy``
  * Install `pandas` using the command: ``pip install pandas``
  * Install `sklearn` using the command: ``pip install sklearn``
  * Install `TextBlob` using the command: ``pip install textblob``

### Generate data.json (output prediction)

- Assuming you have installed all dependencies on the last point, to generate data.json, 2 documents are required TRAINING: `train-data-prepared.json` and VALIDATION: `val-data-prepared.json` whith the following format and at least the following attributes:

```
[
  {
    "id": "id",
    "preceding_posts": [
      {
        "body": "text",
        "controversiality": <value>,
        "violated_rule": <value>
      },
      {
        "body": "text",
        "controversiality": <value>,
        "violated_rule": <value>
      }
    ],
    "label": <1 for ad hominem or 0 not ad hominem>
  }
]
```

Those files must be on the same folder location where `ad_hominem_notebook.ipynb` is located, now, run ` ad_hominem_notebook.ipynb ` and `data.json` will be generated.
** NOTE: Your editor could require to trust in this ` ad_hominem_notebook.ipynb ` file, if so, please trust.

### Goal of the Assignment

- The goal is to train given sentences and to predict given validation data to determine whether the dialog ends in an ad hominem post or not, then, generate an output with all predictions in the following format:

```
{
        "<dialogue_id_1>": 1,
         "<dialogue_id_2>": 0,
        "<dialogue_id_3>": 1
 }
```

### Strategy

To begin with, we cleaned data from training set by deleting punctuation, spaces, url, and stop words, then we created a Pandas DataFrame to keep the important information, such as Labels, Id, preceding posts to have access to all attributes, preceding post with clean data in one string and preceding post with clean data for each post, then we did the same for validation set, after this, we created features and added them into a matrix to send this to the ML Classifier, then we predict values from validation set and finally, we created the output file data.json

### Features

Sentiment:

This feature evaluate if negative sentiment is increasing on the discussion, if second post has more phrases on negative will provide 1, otherwise it will provide 0.
To do so, we use TextBlob sentiment, extracting polarity and evaluating if this is less than 0

Motivation:

We found that when negative opinions increase, an ad hominem will be more likely to appear.

Bad Words:

This feature will search on each post the number of bad words over a pre-defined bad words list, if the amount of bad words increases, then, this will provide 1, otherwise 0, this list was taken from *Maurice Butler https://github.com/MauriceButler/badwords/blob/master/array.js*

Motivation:

We found on the material that introducing vulgar intensifiers is a way to predict if an ad hominem will be following the post.

Controversiality:

This feature will search if any of the topics has the attribute “controversiality” on 1 if so, set 1 otherwise set 0.

Motivation:

We found on the provided material that controversial post may lead to an ad hominen, and after some testing on data we found that from 186 conversations with “controversiality” attribute set to 1, 146 of them ended with Ad Hominem and 40 was labeled as 0, therefore, when this was controversial more than 78% were positive ended with an ad hominem.

Upper Case:

This feature will count the number of upper case words with a length greater than 1 in the whole conversation, we exclude some words that are not relevant to detect ad hominem, such as 'EU' 'OP' 'SF' among others

Motivation:

We found on the provided material that the use of upper case words can detect an ad hominem even 3 post before this happened, phrases like NO ONE CLAIMED THAT

Violeted Rule:

This feature will search if any of the topics has the attribute "violated_rule" and send the value as a feature value.

Motivation:

When the user does not respect the rules or event his comment was removed, violated rule is set with a value greater than 0. Over 16 discussions with "violated_rule" greater than 0, 100% of the post ended with an ad hominem.

TFIDF:

TF-IDF enables us to gives us a way to represents how important each word is in the discussion. After multiples testing with the max features we decided to use 300 words, then, discussions with similar relevant words will have similar vectors, which is what we are looking for in a machine learning algorithm.

Motivation:

It is a classic feature extractor that is used in most text classification problems.

### Modeling Strategy

- We choose Linear support vector machine after running our training and validation data on various classifiers like SVM, Random forest and Neural Networks (RNN), we found that the SVM predicts the posts in the most optimal way because rather than taking a probabilistic approach SVM works on the geometric interpretation of the problems.

- We have tried to model our features with Neural networks like RNN. We created one embedding layer where we passed our TFIDF word embeddings and created another layer for our features like "Bad words, "negative sentiment". We passed these two layers as input to neural. We had one hidden layer where we used activation as "sigmoid". However, the time to train our model was very high and accuracy was also less than the benchmark score of 0.61.

- Thus we decided that using SVM would be a safe bet because unlike other machine learning algorithms they can overfit if they don’t have enough training samples, RNN can also overfit if training goes on for too long - a problem that SVMs do not have. SVM is not prone to catastrophic failures along with that the algorithm is also able to correlate with other elements within the corpus that help to understand the dense features in NLP resulting in sentimental analysis or machine translation, but in the case of other classifiers, the results were not consistent enough.