Assignment 3:
Text classification and Authorship Attribution

In this assignment you will be using various algorithms for two text classification tasks:

1. An authorship attribution (binary) task on Donald Trump's tweets.
2. A multi-class classification of Hebrew COVID-related tweets.

You will have to submit a comprehensive report, along with the code and classification output obtained on a test set.

Submission deadline:
23:59, Thursday, June 29, 2023[1]

This assignment could be submitted in pairs.

Objectives:

1. Learn to work with the sklearn library.
2. Learn to work with the PyTorch library.
3. Learn to work with the Google Colab environment
4. Learn to design and interpret experiments in NLP (authorship recognition, classification).
5. Understand the differences between the various algorithmic frameworks and their application to different types of data.

Algorithms:

You should use Python's nltk, sklearn and Pytorch packages/libraries for preprocessing, training and testing your classifiers (these packages are well documented and usage examples are part of the documentation).

You should use:

1. sklearn.linear_model.LogisticRegression
2. sklearn.svm.SVC (use both linear and nonlinear kernels!)

[1] Note that the deadline is more than a week after the end of the semester and that you have over three weeks to submit this assignment. We understand that the next two weeks may be busy due to the resume of classes so we pushed the deadline in order to allow you to manage your time in a more flexible way. However, we note that the work should not take three weeks and you are encouraged to complete the assignment with the end of the semester.

3. You should use the PyTorch library to build a FFNN classifier (with at least one hidden layer) to achieve the classification. Feel free to experiment with the number of layers (a simple tutorial for FFNN with PyTorch).
4. A fourth classifier of choice (neural or not). You are encouraged to experiment with classifiers that allow combining different types of features (e.g. number of capitalized words, time of tweeting, etc.)
5. A fifth classifier of your choice (this **should be** neural - RNN, or transformer-based) - feel free to experiment.

You are encouraged to use sklearn's cross validation module. Think about the evaluation measures you use.

Part A: Authorship attribution: Who Controls this Account

Politicians, as well as other public figures, usually have assistants and staffers that manage most of their social media presence. However, like many other norm defying actions, Donald Trump, the 45th President of the United States is taking pride in his untamed use of Twitter. At times, during the presidential campaign, it was hypothesized (pdf) that Donald Trump is being kept away from his Twitter account in order to avoid unnecessary PR calamities. Trump's tweets are not explicitly labeled (Hillary Clinton, for example, used to sign tweets composed by her by an addition of '-H' at the end of the tweet while unsigned tweets were posted by her staffers). It is known, however, that Trump was using an android phone[2] while the staffers were most likely to use an iPhone. Luckily, the device information is part of the data available via the Twitter API, hence the device used can be used as an authorship label.

In this task you will be using a number of supervised machine learning classifiers in order to validate the hypothesis about Trump tweeting habits.

Data:

A small dataset containing a couple of thousands tweets from Trump's account posted between early 2015 and mid 2017 can be found in tweets.tsv, available to download here.
The file is in a tab separated format (.tsv), each tweet in a new line. The fields in the file correspond to:
<tweet id> <user handle> <tweet text> <time stamp> <device>

[2] Trump switched to a secured iPhone in April 2017, hence, building an accurate authorship model on older data can be used for authorship attribution of newer tweets.

While the data is already cleaned and filtered, there is still some degree of freedom you will have to take care of. Specifically:

1. The **handle** field: the handle field can take one of the following three user names: realDonaldTrump (this is Trump's account), POTUS (stands for President of the United States, this is the official presidential account, thus not Trump before the election) and PressSec - the official twitter account of the president's Press Secretary.
2. The **device** field: the device field can take various values ranging from 'android', iphone', instagram among other possibilities.
3. The format of the timestamp field is '%Y-%m-%d %H:%M:%S' you can use the *datetime* module and the *strftime()* and *strptime()* functions to parse and process timestamps.

An unlabeled test set with 200 tweets is available here. This file lacks the <tweet id> and <device> fields.

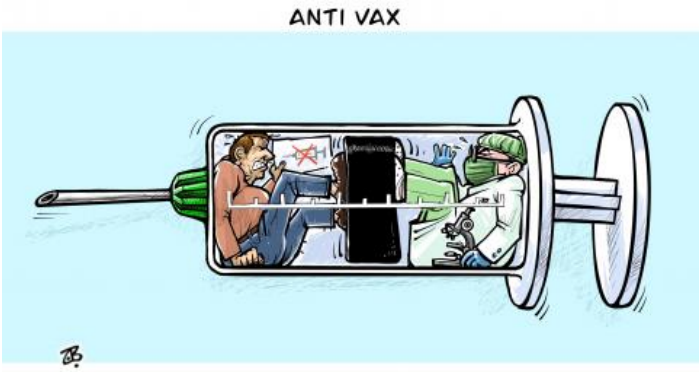
We hold another set (not shared with you) on which we will test your code.

Results

The submitted results file should have a single, space separated line containing only zeros and ones (integers) denoting the predicted class (0 for Trump, 1 for a staffer). The order of the labels MUST correspond to the tweet order in the testset.

Part B: Text Classification - Vaccine Hesitancy

One of the polarizing topics in recent years is the attitude toward vaccines. The debate became heated during the COVID-19 pandemic and the anti-scientific discourse often aligned with political stance (especially in the United States). In this task you will be processing Hebrew tweets, some of them convey an anti vaccination sentiment, some are vaccination related (but not antivaxx) and some are neutral.



Algorithms:

You are required to use five different classification algorithms similarly to the guidelines in the previous task. Note that while some of the pipeline is reusable, this data may require a different

preprocessing. Also note that the data representation may diverge due language difference and the “structure” of the data.

Data:

The training data is a google spreadsheet with two columns: column A contains the text and column B marks the label - 1, 2 or 0, where 2 indicates an anti vaccination attitude, 1 indicated a COVID/vaccine related but no anti vax sentiment is conveyed, and 0 indicates that the tweet is neutral, that is, the tweet is not about the pandemic, although it may reference to it, e.g.,
"מקווה ש illness. זה לא מיסוך עשן לקורונה כי אם כן אז הוא לא משחק גם נגד סיטי"

A training set is available here and a small test set is available here.

Results

The submitted results file should have a single, space separated line containing only zeros, ones, and twos (integers) denoting the predicted class (0 for neutral, 1 for covid-related [not anti-vax], and 2 for anti-vaccination/COVID denial sentiment). The order of the labels MUST correspond to the tweet order in the testset.

Environment, Libraries, and dependencies

You can use Google Colab as your environment (and submit a notebook file). Your colab notebooks should be self-contained with all the necessary imports and should be executed smoothly on Colab. Colab natively supports pandas, numpy, sklearn, nltk and other useful modules and packages. It also works well with Huggingface Transformer models, see examples:

a. A basic example of using a pretrained Roberta sentiment analyzer: here
b. Fine tuning a model: here

Make sure you use supported modules and packages so they can be imported directly, without the need to upload any library.

Report

You should limit your report to three (3!) pages (font size 11p, 1.5 space btw. lines). You can submit your report in Hebrew if you like[3]. The report should be a **PDF** file.

The report should cover both tasks: one page reporting on the Authorship Attribution task, one page reporting on the COVID task and a third page discussing the different performance of the various algorithms on the different datasets (that is - analysis and insights, not just a comparison of numbers that were already provided in the previous pages).

Your report should include a detailed list of models, your assumptions about the data, and the preprocessing steps you took. You should clearly indicate the differences between the different algorithms and the various settings (in each task). Results should be reported in a table.
The third page should include **your insights and conclusions** as learnt from the data.
Specifically you should address the following:

1. How were the corpora preprocessed and why (the preprocessing of each dataset can be described and explained in the respective page. In the "insights page" you are required to reflect on and justify your preprocessing choices.
2. What data/features were used in each setting
3. What is the data representation (input) for each of the algorithms?
4. What are the settings (hyper parameters, etc.) used for each algorithm?
5. Comparison between algorithms and settings.
6. If there are significant performance differences between algorithms/settings - why do you think that is.
7. You should specify the model and the exact parameters that yielded the best results on the test set (as submitted in the results file).

Submission guidelines:

2. You should submit one tar-ball **tar.gz** file with all relevant code, results, and reports. The file should be named `<id 1>_<id 2>.tar.gz` (or `<id>.tar.gz` if you choose to submit alone). The tar-ball should at least include the following files:
    a. **Report**: the main requirement of this assignment is a report file explaining your use of the algorithms and describing the results - comparing results of different algorithms and different configurations of parameters. This file should be named `<id 1>_<id 2>.pdf` (or <your_id>.pdf) .

[3] While the purpose of the report is not to test your English writing skills, we do expect the report to be readable and concise. You will not be punished for mediocre writing but we will reduce points if the report is badly written. We encourage the use of Grammarly or ChatGPT for polishing and rephrasing. If you choose to use assistive tools, please indicate it in a footnote.

b. **A results file for Task A** called `<id 1>_<id 2>_aa.txt` (or `<id>_aa.txt`) corresponding to this test set. This file should hold the results of your best performing model. Format specification above.
c. **A results file for Task B** called `<id 1>_<id 2>_covid.txt` (or `<id>_aa.txt`) corresponding to this test set. This file should hold the results of your best performing model. Format specification above.
d. **Source code**: You should submit a Python Notebook (`ex3_<id 1>_<id 2>.ipynb`) that could be uploaded and executed in Google Colab.
Code files should be well documented with clear usage examples so different models could be easily executed. You should base your notebook on this skeleton (make a copy and rename)and it should support the following API (see documentation in the notebook):

```
def training_pipeline(task, alg, train_fn)
retrain_best_model(task)
def predict(m, fn, task='aa')
def who_am_i()
```

3. While we are not focused on code optimization, we do expect your code to run in a reasonable time. That is - if your code runs for a couple of hours it may suggest something is wrong.