Problem Set 2

Due Date: 11:59pm on **Oct 31, 2022**

Submission: You will need to make two submissions:

- Please submit all code that you wrote and the output of your scripts (including diagrams) as a zip file (your_kerberos.zip) under the "Problem Set 2 Code" assignment on Canvas.
- Additionally, please submit a written report (your_kerberos.pdf) under the "Problem Set 2" assignment on Canvas. The submission will be uploaded through Gradescope, where you will be able to select the individual pages in your report corresponding to each problem.

Late Submission: Late assignments will not be accepted without documentation of a valid emergency from S3/gradsupport for MIT students, or an equivalent student support hub for non-MIT students.

Collaboration: You may collaborate with your fellow classmates (and in fact, you are encouraged to!). However, what you turn in must be your own version of the work. If you do collaborate with some of your classmates, please make sure to list your collaborators on the first page of your submitted report.

Grading: Make sure that your solutions to the problems are written clearly and concisely. You will be graded both for answering the questions correctly and for writing up your answers in a readable manner.

GitHub Repo: Unless otherwise noted, all scripts needed for this problem set can be found at: `https://github.com/criticaldata/hst953-2022`

Problem set prepared with help from Adrien Carrel, Vassili Chesterkine, and Arnaud Petit

Problem 0. Preliminaries [0 points]

In this assignment, you will be working with the MIMIC-III dataset (`https://mimic.mit.edu`) which consists of about 40,000 patients admitted to ICUs in the Beth Israel Deaconess Medical Center in Boston between 2001 and 2012.

Follow the instructions at the following link `https://mimic.mit.edu/docs/gettingstarted/cloud/` to get access to the MIMIC-III dataset on Google BigQuery. You will need BigQuery access to complete the homework assignment.

Problem 1. Modeling with ClinicalBERT Embeddings [15 points]

Most of the datasets we have seen so far in this course have been cleaned and preprocessed. However, in some situations, the datasets available to us might not have a true ground-truth label for us to use. In this problem, we will explore using the ClinicalBERT language model to assign labels to datapoints based on the natural language ICU discharge summaries corresponding to each datapoint.

(a) [**1 pts**] The language model we will use for this problem is ClinicalBERT, a language model initialized from BioBERT (a bi-directional transformer trained on biomedical text data) and fine-tuned on MIMIC-III clinical notes.

You can read more about ClinicalBERT here: `https://huggingface.co/emilyalsentzer/Bio_ClinicalBERT`, and after doing so, fill out the code block in the notebook to initialize the ClinicalBERT model and tokenizer using the huggingface library.

(b) [**1 pts**] To get used to working with the huggingface pipeline, follow the instructions in the notebook to fill out the `fill_blank` function.

The function should take as input a sentence containing a missing word (denoted by an underscore) and return that same sentence but with the missing word replaced by the most likely token according to ClinicalBERT.

Ensure that your implementation returns the expected results on the test sentences before proceeding to the next part.

(c) [**2 pts**] As was hinted at by the test sentences, even though ClinicalBERT was trained and fine-tuned on only biomedical and clinical texts, it still exhibits some societal biases. Find another pair of sentences, similar to the test sentences, that exhibits another kind of societal bias. Include those sentences with their ClinicalBERT completions in your report.

(d) [**1 pts**] What is one outcome (on label-assignment accuracy) that might result from the use of biased language model representations to automatically extract "ground-truth" labels for a dataset? Briefly explain your answer.

(e) [**1 pts**] So far we have only worked with the *probabilities* assigned by ClinicalBERT to a particular word or token in a sentence. However, ClinicalBERT also has a "hidden state representation" corresponding to each word or token in the input sentence. It is usually these representations that are used to perform downstream tasks such as automated datapoint labeling, so as a first step, follow the instructions in the notebook to extract a representation, or "embedding", for the discharge summary corresponding to each datapoint in the dataset. Make sure to save the embeddings to your drive to avoid needing to compute them every time you run the notebook.

(f) [**1 pts**] One way of checking how rich the representations are is to train a linear model on them to perform the desired downstream task. The reasoning behind this is that on a complex enough dataset, a linear model will not be powerful enough to capture all of the complexities of the underlying data, so if it is able to perform well on the downstream task, then that means most of the "heavy-lifting" was done by the language model when computing its hidden representations.

Follow the instructions in the notebook to train a logistic regression model on the computed embeddings to predict whether each patient in the dataset had hypertension during their stay in the ICU.

Note that in an automated label-extraction setting we wouldn't have access to any hypertension ground-truth labels. However for our purposes, we are merely investigating the suitability of using language model representations to extract the desired labels.

(g) [**1.5 pts**] Compute the accuracy, precision, recall, and F1-score on each of the training and test sets. For precision, recall, and F1-score, compute the *per-class* metrics (i.e. with `average=None`). How well does your model perform on each class of patients: those who have hypertension and those who do not?

(h) [**2 pts**] For each of the training and test sets, use UMAP to reduce the dimensionality of the data down to two dimensions and plot the resulting data on a scatter plot (one scatter plot for the training set and another one for the test set). Use different colors for the patients who had hypertension and those who did not.

Vary the values of the `n_neighbors`, and `min_dist` parameters and include the best resulting plots in your report. A brief description of the parameters can be found at the following link: `https://umap-learn.readthedocs.io/en/latest/parameters.html`.

Note: by "best" resulting plots in this part, we mean the plots that separate the two classes as much as possible.

(i) [**1 pts**] From the UMAP plots, do the classes look linearly separable? Do the plots support the logistic regression model's hypertension prediction results? Briefly justify your answer.

(j) [**1.5 pts**] While UMAP is very good at projecting high-dimensional data down to a few dimensions, it does not make use of class label information to try to maximally separate the two classes in its projections.

Follow the instructions in the notebook to project the data down to one dimension using Linear Discriminant Analysis. The key idea behind LDA is that it projects the data down onto the dimension(s) that maximally separate the target classes.

For each of the training and test sets, generate two histograms from the LDA projections: one for patients who had hypertension and one for patients who did not, and plot both histograms on the same plot. (So in total you should have two plots: one for the training set, showing two overlapping histograms, and one for the test set, showing two overlapping histograms). Include the plots in your report.

(k) [**1 pts**] From the histogram plots, do the classes look linearly separable? Do the plots support the logistic regression model's hypertension prediction results? Briefly justify your answer.

(l) [**1 pts**] Do ClinicalBERT embeddings seem like a good choice for automatic label extraction for hypertension? Briefly justify your answer.