Problem set 1

This problem set is due Thurs Feb 21 at 11:59pm EST. Please submit your write-up and code. When you write up your report, put all writing into one file and name it ${mit_email_username}.pdf (e.g. iychen.pdf). Please note any and all collaborators. You may not share code or write-ups with anyone.

We are interested in predicting ICU mortality from clinical records. At this point, you should have access to the MIMIC-III dataset and to Physionet worksplace. If you do not, please refer to Problem Set 0. If you still have problems with access to the data, please contact the course staff through Piazza.

Although all data comes from MIMIC-III, we have parsed it for your convenience. You may download our derived data using our class workspace on Physionet. Other code is available on the Github repo.

For consistency, we refer to patients with `train=1` as training data, patients with `valid=1` as validation data, and patients with `test=1` as test data.

1. Data exploration

The first step when accessing a new dataset is take a look and understand the dataset.

We hope that you find this exercise useful in understanding the clinical care received by patients. When everything is just dataframes and vectors, it might be easy to lose sight of the fact that we are trying to use data to help real people with serious problems.

In the notebook `chart_review.ipynb`, we will walk through one patient's hospital course together. Then you will analyze a second patient's course and include that description in your pset writeup.

1.1) What is the patient's History of Present Illness?

1.2) How long were they in the hospital?

1.3) What are some of the major events in the timeline of this patient's care?

1.4) What stood out to you most when reading the nursing notes?

2. Structured data

We have provided the first-collected lab results from the first 48 hours of a patient's stay. Take a look through the data of `adult_icu.gz` and the associated code that generated it `mort_icu_cleanup.py`.

2.1) What do the columns seem to mean? If we wanted to predict hospital mortality, which column would we use as the outcome column? Which columns would be features and which columns would NOT be

features?

Using a logistic regression – we recommend sklearn – train a classifier on the training data while optimizing for best train accuracy, validate the best hyperparameters (`C=[0.1,0.25,0.5,1.]` and `penalty=['l1','l2']`) on validation data.

2.2) What is the accuracy on test data? What is the AUC?

2.3) What are the most predictive features? Comment on your findings.

3. Clinical notes (logistic regression)

Clinical notes can be a rich source of unstructured information. For ease of computation, we randomly selected 15,000 patients from the dataset and supplied the notes from the first 48 hours in `adult_notes.gz`.

Use a count vector of a bag of words of the 5000 most popular words (i.e. the 5000 words with the highest term frequency) and a logistic regression trained to optimize the data accuracy. Note that this prediction task is only using the clinical notes and not using any structured data from the previous section.

Train on training data and validate the best hyperparameters (`C=[0.1,0.25,0.5,1.]` and `penalty=['l1','l2']`) on validation data.

3.1) What is the accuracy and AUC on the test data? How do these results compare to structured data results?

3.2) What words are the most predictive (positive and negative) of ICU mortality? Look up any unknown clinical definitions and comment on a few.

4. [optional, not graded] Clinical notes (CNN)

Given recent advances in deep learning, we are curious as to whether we can beat the performance of a logistic regression.

Since training a convolutional neural network (CNN) to predict hospital mortality can be computationally expensive (~8 hours on a normal laptop), we have provided you with a pre-trained model and sample code in `p4.py` to load it. Note that loading the pretrained model and creating the vocabulary may take awhile (~30 minutes on a normal laptop). You may also need to install multiple packages.

4.1) Take a look at the model architecture in `p4_model.py`, particularly the class `CNN`. Describe your model architecture. How many layers are there? Which filters do you use? How are you aggregating the layer outputs?

Using the code in `p4.py`, make predictions for the test data. Note that you will need to create separately `train.csv`, `valid.csv`, and

`test.csv` files from `adult_notes.gz` file with fields `['icustay_id','chartext', 'mort_hosp']` in that order.

4.2) What is the test accuracy of your model? Comment on the performance compared to Part 2.

We are interested in the cases where the CNN predicts better than the logistic regression (LR) for clinical notes. For how many patients does your CNN and LR disagree? Using the `icustay_id` to unify records, use the structured data from Part 2 to compare a) the population of patients where the CNN predicts correctly but the LR does not and b) the general dataset population.

4.3) Specifically, what does the age distribution look like across both groups? When comparing a LR trained on a bag of words and a CNN with more complex filters, what are potential explanations for these observed findings?