Mini Project on Sequence Modeling

Project Structure

In this individual mini-project, you will practice some of the sequence modeling skills that you learned in Assignment 4 by applying them to a new dataset. As you may recall from Assignment 4, many sources of data include a temporal (time-based) aspect to them. For example, word order is often essential to the meaning of a phrase. Or, a set of descending musical notes gives a different feeling than an ascending one.

We can apply sequence information for various tasks, for example:

- Predicting if a coin is fair or not
- Choosing the next song to play in a playlist based on previous songs
- Deciding what to suggest when watching videos
- Determining if text came from one corpus or another
- Predicting the next word in a phrase
- Writing a negative movie review based on a few seed words :)
- Creating a word completion tool based on starting letters.

In this mini-project, you will:

1. Choose a data set that involves sequence information (see below for some example data)
2. Apply n-grams to your data. Optionally, you may also choose to explore and apply other sequence representations, if you are excited about learning new tools in this space. We have included a few resources to lightly support this at the end of this document.
3. Use the n-grams (or another representation) to do something interesting with your data. Some examples of this could include:
    a. Predicting the next item in a sequence
    b. Classifying which corpus a particular sequence came from
    c. Topic modeling (e.g., using Latent Dirichlet Allocation, which is built into sklearn)
    d. Visualizing representations of your data
    e. Estimating if a dataset fits a particular type of model (like we did with the coin flip example… does this sequence of coin flips come from a fair coin?; One could use this to look at some model of fairness.)

We hope this is an opportunity to creatively apply and practice some techniques in any application that you are excited about.

Expectations and assessment

This assignment spans a full week of class, starting Thursday and submitted next Thursday. Our expectation is that you will spend 7-9 hours actively working on this mini-project (this is fewer than the typical 12 hours per week on average for Olin courses, since Monday will be devoted to other non-course activities).

Please submit:

- A link to your notebook: We will not provide direct feedback or assessment of your notebook, but we are very excited to look at them.
- One slide about your project (here for morning, and here for afternoon).
- A short reflection on your learning process and constructive engagement
- A self-assessment score on a scale of 0 to 15. This mini-project will count as 1.5 regular assignments. Our mental model of self-assessment here aligns with your regular assignment submississions: did you fully engage with this assignment?

Potential Datasets

More Scaffolded Option

If you want to spend a bit less time with data wrangling and more time fitting your models and experimenting, you may consider using the All the News Dataset. We put together a notebook that loads the data, tokenizes it into words, and runs it through sklearn's vectorizer. Additionally, since this is a Kaggle dataset you can take advantage of the Kernels page to see code that others have written to analyze the data (it won't all be necessarily in the same vein as what we've been doing or understandable, but it could provide some nice inspiration).

Other Options

Here is a list of individual datasets as well as some searches on dataset repositories that we think could be interesting. We haven't vetted these in much detail, so please investigate them before sinking too much time into them.

- OpinRank Review Dataset Data Set
- OpenML has a wealth of datasets. Here are some search terms and a link to how to download them using sklearn.
  - https://www.openml.org/search?q=text&type=data

    - https://www.openml.org/search?q=review&type=data (beer review dataset?!?)
    - Fetch openml data with sklearn

- Cyber Trolls Dataset (aggressive versus non-aggressive tweet classification).
- Russian Trolls Twitter Dataset (see FiveThirtyEight article Why We're Sharing 3 Million Russian Troll Tweets)

- Kaggle has a huge wealth of datasets. It also has the advantage of having Kernels for some of them (which provide sample code). Here is a search we did for the term "text". https://www.kaggle.com/datasets?search=text
- Google has shared some of their n-grams: https://ai.googleblog.com/2006/08/all-our-n-gram-are-belong-to-you.html

- If you want to do something other than text you might consider audio (get in touch with us if you want to go this route). There is a model called bag of audio words that converts sound into an auditory alphabet, which allows you to apply many of the same sequence learning algorithms you've used for text.

Potential Things to Try

A sensible default for the project would be to apply ngrams + Naive Bayes to classify the sequence (e.g., some text) along some dimension (e.g., spam versus non-spam, positive sentiment versus negative sentiment). If you go this route, you may want to consider layering on some analysis of the model you learned.

- What words are most associated with predicting a particular output (we'll provide some sample code for this over the next day or so)
- Did the model learn any biased associations (e.g., in the Amazon resume evaluation system some words like "women's" or the names of some women's colleges were negatively associated with selection for an interview).
- How do hyperparameters (such as alpha) affect the performance?
- How does the choose of *n* when using ngrams contribute to performance.

If you want to try something off the beaten path, here are some suggestions.

- Topic modeling is a way of clustering words that seem to co-occur with a collection of documents. The most popular way of doing this is to use an algorithm called Latent Dirichlet Allocation. You probably have enough background on Bayesian methods to get the gist of the model, but probably don't have enough background to get all the details. That said, you can certainly learn about it in a top-down fashion by using sklearn's implementation (here is an example, skip the parts about NMF)

- Explore word2vec. You can transform your text tokens into vector representation using word2vec (which you read about). Please reach out if you go this route so we can help you. Here is one walkthrough of using this for text classification.