ECE 562 Machine Learning, Fall 2019
Instructor: Zafer Aydın
Computer Homework 7

Introduction

Download the template codes for Udacity's Intro to Machine Learning course by running the following command in a terminal

git clone https://github.com/udacity/ud120-projects.git

Submit your scripts as q1.py and/or a text document that includes your answers to the questions below through Canvas.

Assignment

1. You'll start with a warmup exercise to get acquainted with parseOutText(). Go to the tools directory and run parse_out_email_text.py, which contains parseOutText() and a test email to run this function over.

parseOutText() takes the opened email and returns only the text part, stripping away any metadata that may occur at the beginning of the email, so what's left is the text of the message. We currently have this script set up so that it will print the text of the email to the screen, what is the text that you get when you run parseOutText()?

2. In parseOutText(), comment out the following line:

words = text_string

Augment parseOutText() so that the string it returns has all the words stemmed using a SnowballStemmer (use the nltk package, some examples that I found helpful can be found here: http://www.nltk.org/howto/stem.html). Rerun parse_out_email_text.py, which will use your updated parseOutText() function--what's your output now?

Hint: you'll need to break the string down into individual words, stem each word, then recombine all the words into one string

3. In vectorize_text.py, you will iterate through all the emails from Chris and from Sara. For each email, feed the opened email to parseOutText() and return the stemmed text string. Then do two things:

remove signature words ("sara", "shackleton", "chris", "germani")

append the updated text string to word_data -- if the email is from Sara, append 0 (zero) to from_data, or append a 1 if Chris wrote the email.

Once this step is complete, you should have two lists: one contains the stemmed text of each email, and the second should contain the labels that encode (via a 0 or 1) who the author of that email is.

Running over all the emails can take a little while (5 minutes or more), so we've added a temp_counter to cut things off after the first 200 emails. Of course, once everything is working, you'd want to run over the full dataset.

vectorize_text.py can be found in the text_learning directory

What is the string that you get for word_data[152]?

ECE 562 Machine Learning, Fall 2019
Instructor: Zafer Aydın
Computer Homework 7

4. Transform the word_data into a tf-idf matrix using the sklearn TfIdf transformation. Remove english stopwords. Be sure to use the tf-idf Vectorizer class to transform the word data. Don't forget to remove english stop words when you set up the vectorizer, using sklearn's stop word list (not NLTK).

You can access the mapping between words and feature numbers using get_feature_names(), which returns a list of all the words in the vocabulary. How many different words are there?

Remove the code for temp_counter and consider all possible words. How many different words are there?

What is word number 34597 in your TfIdf (i.e. the word indexed as 34597)?