Announcements
● Homework 10 due Thursday

Outline
● Warm-up
● Sentiment analysis
● Machine learning
● Training, applying, and testing a word sentiment model

Warm-up
Write a function called `get_words` that takes a string as a parameter and returns a list of words in the string. Only digits and letters should be kept (all other characters should be ignored). All letters must be converted to lowercase.
High-level steps:
● Initialize accumulator variables representing the current word and a list of words
● Loop over all characters in string
  ○ If character is space, then add current word to list and reset current word
  ○ If character is digit or letter, then add character to current word

Sentiment analysis
● Varying depths of analysis
  ○ Simple: is the attitude of the text positive or negative?
  ○ More complex: rank the attitude on a scale of 1 to 5
  ○ Advanced: identify the source, target, and type of the attitude
    ■ Label the targets and types in the following review.
      “Definitely the best pizza in town. And the people who work there are kind, patient, and helpful.”
      type    target     target     type
          type
● High-level steps
  ○ Tokenization --- breaking text into words, numbers, emoticons, etc.
  ○ Feature extraction --- Which words to use? only adjectives? all words?
  ○ Classification --- assign a label (e.g., positive, negative, neutral) to the text
    ■ Requires a model --- a set of rules or mathematical functions
    ■ Models are often constructed automatically using machine learning
● Challenges
  ○ Handling negation --- e.g., “I don’t like this pizza” vs “I really like this pizza”
    ■ Solution: “distribute” negation to words the follow
  ○ Sarcasm --- e.g., https://www.youtube.com/watch?v=aT8UhqNyhB4
    ■ Hard for both humans and computers (https://dl.acm.org/citation.cfm?id=2002850)
    ■ Partial solution: nearest neighbor (http://www.aclweb.org/anthology/W10-2914)
Machine learning

- Computers “learn” from data, rather than being explicitly programmed
- Example
  - Given a set of restaurant reviews that are manually labeled by a human as positive or negative
  - Goal: learn whether each word (e.g., “fantastic”) conveys positive or negative attitude
- Many types of machine learning algorithms
  - Decision trees
  - Linear regression
  - Naive bayes --- use a simplified version of this for homework 10
  - Nearest neighbor
  - Neural networks
- Three phases
  - Training --- learn from a set of sample data that has a known outcome
  - Testing --- apply the model to a different set of sample data that has a known outcome
  - Application --- apply the model to data with an unknown outcome

Training a word sentiment model

- Goal: learn the sentiment of a word (positive, negative, or neutral)
- Input: a set of reviews that include both written comments and numeric ratings (e.g., 1 to 5 stars)
  - Called the training set
  - Example training set
    5 This product is good.
    5 This product is made very well.
    4 This product functions well.
    3 This is a mediocre product.
    2 A piece broke off. I am dissatisfied with this product.
    1 This product is junk.
    1 This product does not function as it is advertised.
- Output: numeric rating (e.g., between 1 and 5) for each word appearing in at least one review
  - Called the model
- **How do we compute a word’s numeric rating?**
  - Sum the numeric ratings of the reviews in which the word occurs and divide by the total number of occurrences. If the same word appears more than once in a single review, then we count the word multiple times.
- **Write a function called train that takes a list of tuples as input and returns a dictionary of words and their numeric rating. Each tuple contains a numeric rating (an integer) and a list of words (i.e., a list of strings).**
- High-level steps:
  - Initialize two empty dictionaries for storing words (keys) and the corresponding sum of review ratings and number of occurrences (values)
  - Loop over all tuples
    - Loop over all words
      - Add review rating to sum of review ratings for word
      - Add one to number of occurrences of word
    - Divide sum by number of occurrences
  - Return dictionary of words (keys) and their average rating (values)
Applying a word sentiment model

- Write a function called `compute_score` that takes a list of words (i.e., a review) and a dictionary of word ratings (i.e., the output from `train`) and returns an estimated rating for the review.

High-level steps:
- Initialize accumulator variables (total and count)
- Loop over all words
  - If word in dictionary
    - Add word’s numeric rating to total variable
    - Add one to count
- Return total divided by count