Project overview
Matt Langsenkamp, Eben Carek, Joseph Sebastian

Main Idea

The project idea is to design a method for augmenting training data for sentinment analysis. This will be done by leveraging word vectors and a couple of assumptions.

Assumption 1: Two words with similar word vectors are synonyms.
Assumption 2: Within the word vector space the nearest neighbor of a word is its strongest synonym.
Assumption 3: If any sentence from the original training data is taken, and any amount of words are swapped with their respective strongest synonym, the resulting sentence will still have the same sentiment classification.

High level roadmap of experiment

1

Obtain suitable dataset of size N, with labels for sentiment analysis (ie Imdb).

2

Preprocess original dataset(ie bag of words)
Feed original data to classifier(ie naive bayes)
report accuracy.

3

Generate word vectors from original dataset (ie GLoVe).
Generate augmented dataset useing word vectors. Preprocess augmented dataset(ie bag of words)
Feed augmented data to classifier(ie naive bayes)
report accuracy.

Augmentation method discussion

There are a couple different ways that a dataset could be augmented or duplicated. Each method will be implemented and tested using the experiment framework above. A high level psuedocode and explanantion of each method is provided below.

Method 1 Takes each sample and makes a new sample with a random amount of words that have been swapped with there respective synonym.

```
augmented_data = []
for sample in dataset:
  new_sample = sample.copy()
```

```
  indicies = pick_word_indicies(sample)
  for indicie in indicies:
    new_sample[indicie] = get_synonym(sample[indicie], vector_space)
  augmented_data.append(new_sample)
  augmented_data.append(sample)
```

Method 2 Takes each sample modifies it by randomly choosing words and then appending there synonym after it. For example if the sample was "isn't it great here" and "great" was the chosen word and its synonym was "good". the resulting sentence would be "isn't it great good here"

```
augmented_data = []
for sample in dataset:
  new_sample = sample.copy()
  indicies = pick_word_indicies(sample)
  for indicie in indicies:
    new_sample = sample[:indicie] + get_synonym(sample[indicie], vector_space) + sample
    [indicie + 2:]
  augmented_data.append(new_sample)
```

Method 3 pretty much the same as method one except the only words that can modified are those that are identified as verbs, or adjatives.

```
augmented_data = []
for sample in dataset:
  new_sample = sample.copy()
  indicies = pick_word_indicies_verb_or_noun(sample)
  for indicie in indicies:
    new_sample[indicie] = get_synonym(sample[indicie], vector_space)
  augmented_data.append(new_sample)
  augmented_data.append(sample)
```