Sentiment Analysis

classify imdb reviews as POS or NEG.

Project 1:

```
from collections import Counter
positives = Counter()
positives[token] += 1 → used like a dict()
positives.most_common()
```

Project 2:

allocating memory is expensive. It's better to allocate less frequently. np.zeros((1, vo-s))

Project 3:

- it's useful to monitor "correct so far" samples as the model trains.
- lowering the learning rate might be useful if the network is not learning anything.

Project 4:

- Weight initialization strategy matters!
- one strategy: np.random.normal
- a rule of thumb:

  it's good practice to start weights in range $[-y, y]$ where $y = 1/\sqrt{n}$.

  (n: number of inputs to a given layer)

- noise versus signal in NN;

  value of the inputs of the network, affects weights a lot. If one of them is 18, and the other is 2, the one with value = 18 will dominate highly.

- when we tokenize the data, we should look out for meaningless stuff such as punctuation and common / filler words. Because they are not contributing to the prediction.

  (i.e, noise)

Project 5:

- How to improve efficiency of computation?

  i.e., what is wasteful in the network?

  ① long input vector: if many of them are zero, we're doing a useless mat mul.

  ② if we have "2" in the input vector, we should not do mult.

Project 6:

if we strategically reduce vocabulary size, we may improve the accuracy.