Sentiment Analysis

classify imdb reviews as POS or NEG.

**Project 1:**
```python
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, 100, 5))

**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: \( \frac{\text{random normal}}{\sqrt{n}} \)
- A rule of thumb:
  
  It's good practice to start weights in range \([-y, y]\) where \(y = \frac{1}{\sqrt{n}}\).

- 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 matmul.
  
② if we have "1" in the input vector, we should not do matmul.

Project 6:

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