



# Task

- Run segmentTest using the unigramWeight of 0.5 instead 
- What are the new sequences and their maximum likelihoods? Explain why the results change 

# Solution

Before: unigramWeight = 0.01

Output =

```
[the, real, deal] -3.6222183217678037
[is, it, over, yet] -4.48690799038008
[is, playing, now] -4.929028367185977
[this, is, who, we, are] -4.033755586538169
[a, really, good, job] -3.7384044382284043
```

After: unigramWeight = 0.5

Output=

```
[the, real, deal] -2.318210653291755
[i, sitover, yet] -2.246299183193592
[i, splaying, now] -1.3790286168333126
[this, iswhowe, are] -1.6349911984525107
[are, ally, good, job] -1.1878069467799135
```

Differences: For this set of sample segments, the performance was worse with unigramWeight = 0.5 as it did not segment the words correctly.

## Change in segments

With a higher weight on unigrams (going to .5 from 0.01), we are discounting less than before the probability associated with a unigram. Why is this important – because the backoff method we are using to calculate probabilities turns to unigrams when a bigram (conditional) probability doesn't exist. When does this occur? Mostly when analyzing weird "word" combinations like "yg" "oodjob". Therefore it makes sense that when we increase unigramWeight, we increase the chance of smaller more popular segments/words that stand alone, in our case "I" becomes more probable than "is" because of the weight giving a higher than before probability to unigrams.

## Change in probabilities

With an increase in unigramWeight, we increase overall probabilities of almost all possible sequences.

Added note: Before when unigramWeight was 0.01, you seriously handicapped the probabilities of word pairs that didn't have an existent bigram in the sample txt file. This ups the relative probabilities of whole words (seen at least once in the txt file) compared to "words" not seen before that in the sample data that resorted to unigrams.