Senseval Decision List
Classification

24 February 2022
Words can have different meanings or *senses*.

The challenge of word sense disambiguation is to identify *which* sense a particular word token corresponds to.
This week we’re taking part in a shared task.
First, we step into the time machine.
Senseval 3 in 2004 had participants identify noun and adjective senses from WordNet verb senses from the Wordsmyth dictionary
Do you know what it is, and where I can get one? We suspect you had seen the Terrex Autospade, which is made by Wolf Tools. It is quite a hefty spade, with bicycle-type handlebars and a sprung lever at the rear, which you step on to activate it. Used correctly, you shouldn't have to bend your back during general digging, although it won't lift out the soil and put in a barrow if you need to move it! If gardening tends to give you backache, remember to take plenty of rest periods during the day, and never try to lift more than you can easily cope with.
Do you know what it is, and where I can get one? We suspect you had seen the Terrex Autospade, which is made by Wolf Tools. It is quite a hefty spade, with bicycle-type handlebars and a sprung lever at the rear, which you step on to <head>activate</head> it. Used correctly, you shouldn't have to bend your back during general digging, although it won't lift out the soil and put in a barrow if you need to move it! If gardening tends to give you backache, remember to take plenty of rest periods during the day, and never try to lift more than you can easily cope with.

Everything inside the **lexelt** tag is about a single lexical element (a word with its part of speech)
Do you know what it is, and where I can get one? We suspect you had seen the Terrex Autospade, which is made by Wolf Tools. It is quite a hefty spade, with bicycle-type handlebars and a sprung lever at the rear, which you step on to \textbf{activate} it. Used correctly, you shouldn't have to bend your back during general digging, although it won't lift out the soil and put in a barrow if you need to move it! If gardening tends to give you backache, remember to take plenty of rest periods during the day, and never try to lift more than you can easily cope with.

Each lexical element has multiple instances, which are our training examples.
Do you know what it is, and where I can get one? We suspect you had seen the Terrex Autospade, which is quite a hefty spade, with bicycle-type handlebars, which you step on to activate it. Used correctly, you shouldn't have to bend your back during general digging, although it won't lift out the soil and put in a barrow if you need to move it! If gardening tends to give you backache, remember to take plenty of rest periods during the day, and never try to lift more than you can easily cope with.

There's a unique ID for each instance, and we identify the corpus it came from.
Do you know what it is, and where I can get one? We suspect you had seen the Terrex Autospade, which is made by Wolf Tools. It is quite a hefty spade, with bicycle-type handlebars and a sprung lever at the rear, which you step on to activate it. Used correctly, you shouldn't have to bend your back during general digging, although it won't lift out the soil and put in a barrow if you need to move it! If gardening tends to give you backache, remember to take plenty of rest periods during the day, and never try to lift more than you can easily cope with.

This identifies the correct sense for the word as it's used in the passage – Wordsmyth sense 38201
While the training data marks the correct sense(s) in the XML, the test data doesn’t.

Therefore, we won’t use the `<answer>` tags; we’ll use an external file that has the annotations in it, e.g.,

```
activate.v activate.v.bnc.00044852 38201 38202
```
It’s not a major goal for this course for you to learn to process XML, so you’re given the file `Lexelt.py` that does this parsing for you.
In Part 1, you’ll explore this data set.
What we’re doing is a machine learning problem:

Given training data that’s annotated with the right labels, we learn to predict the right sense of a word.

We do this prediction for the test data and then we look at the annotations for these instances to find out if our prediction is right.
For lots of machine learning problems, there are standard algorithms, which are implemented in Python libraries.

One popular library is scikit-learn. Each of its classifiers has the same interface:

We train the classifier by running `fit(X, y)`

where $X$ is the training data and $y$ are the correct labels for them.

We predict labels by running `predict(X)`
What is X?

It’s not the training data itself; it’s features that we find in that data.

A feature can be anything you compute from the data, e.g.,

- the number of word types
- the number of word tokens
- whether the text includes the word "apple"
For this assignment, you’ll compute two types of features:

- bag of words
- collocation
A *bag of words* is just counting how many times each word – or other token like punctuation – occurs in the text (ignoring the *order*).
*Collocation features* are n-grams that include the head word (the word we’re predicting the sense of).
… upon which parts of the sensory system are activated: stimulation of the retinal receptors …
... upon which parts of the sensory system are activated: stimulation of the retinal receptors ...
To get a better understanding of how this prediction works, we’ll be *implementing* a classifier rather than using one of scikit-learn’s.
A decision list classifier looks at the features we compute and asks how likely a sense is given those features.

For example, if our bag of words includes high counts for *deposit* and *loan*, it’s more likely to be *bank* as in financial bank rather than *bank* as in riverbank.

How do we know that? Maximum likelihood estimation!
Maximum likelihood estimation very simply means that we use the counts from the training data to compute the probabilities for the senses.
\[ \text{score}(f) = \log_2 \frac{P(\text{sense}_1 \mid f)}{P(\text{sense}_2 \mid f)} = \log_2 \frac{\text{count}(f \text{ is present in sense}_1)}{\text{count}(f \text{ is present in sense}_2)} \]
Minor complications:

We’re not just choosing between two senses – there may be any number.

We need to do some smoothing to handle unseen data.
score(f) = \log_2 \frac{P(\text{sense}_1 \mid f)}{P(\text{sense}_2 \mid f)}

= \log_2 \frac{\text{count}(f \text{ is present in sense}_1)}{\text{count}(f \text{ is present in sense}_2)}

score(i, f) = \log_2 \frac{\text{count}(f \text{ is present in sense}_i) + \alpha}{\text{count}(f \text{ is present all other senses except sense}_i) + \alpha}
We train the decision list classifier by:

- Calculating a score for each (feature, sense) pair.
- Sort so the highest scoring features are first.
- Throw away unhelpful scores.
We predict using the decision list classifier by:

Going through the (feature, sense) pairs until we find a feature that matches.

This is the most predictive feature we found!
Label the word as the corresponding sense.
Acknowledgments

The lecture incorporates material from:

Julie Medero, Harvey Mudd College
Richard Wicentowski, Swarthmore College
Xanda Schofield, Harvey Mudd College