Mimimum Description Length (MDL)

Miles Osborne

School of Informatics
University of Edinburgh
miles@inf.ed.ac.uk

March 11, 2008

1 Machine learning

2 Occam's Razor

3 MDL Principle

Learning

As we saw in the language model and in the vector-based machine learning classes:

- We can use a *training set* to *estimate* probabilities.
- We also use our training set to tell us the *model structure*:
  - For a language model, this is the set of ngrams.
- Often, we have many possible choices for our model structure.
- Which one is best?

Learning

We can use Bayes Theorem to give us an answer:

- Call the training set D.
  - D will consist of a set of *examples* $d_1 \ldots d_n$.
  - With language modelling, each sentence could be an example.
- Call our model M.
  - Using our language example, the model will consist of the set of ngrams and corresponding probabilities.

Learning

We search for a good model which is best predicted by the data:

$$M^* = \text{argmax}_{M^*} P(M^* \mid D)$$

$$= \text{argmax}_{M^*} \frac{P(D|M^*)P(M^*)}{P(D)}$$

$$\propto \text{argmax}_{M^*} P(D \mid M^*)P(M^*)$$

This breaks learning down to two tasks –model the training data well, using a likely model

Learning

Modelling the training data:

- The probability of the training set is as follows:

$$P(D \mid M) = \prod_{i=1}^{n} P(d_i \mid M)$$

- This assumes independence between sentences.
- (We know from the language model class how to compute each sentence probability –here we have just explicitly mentioned the corresponding set of ngrams).

Learning

What makes for a good model?

- It should explain the training data well:
  - Good training examples should receive a high probability.
  - Bad training examples should receive a low probability.
- It should also assign high probabilities to *unseen* sentences.

Modelling the training data

As we saw from the LM tutorial, unigram models give a low probability to the training set:

- The higher the ngram order, the better we model the sentences.

This suggests that we could use the full joint probability expansion for our ngrams:

$$P(w_1, w_2, \ldots, w_n) = P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1, w_2)\ldots$$

This will assign a high probability to the training data (. . . but)

Occam's Razor

We know that modelling the training data *too well* can be bad.

- Instead, we want to assign preferences to certain models.
- There are many ways to state a preference for models.
- A popular one is *Occam's Razor*:

Occam's Razor

All else being equal, select the simplest theory possible, but ensuring it is not too simple.

- Occam's Razor has an important place in Science.

This tells us to look for a simple model, but one which is not empty.

Applying the Razor

How can we apply Occam's Razor?

- We know from our entropy and coding classes that we can *encode* items.
- $l(x)$ gives the bit length of the code for item x.
- We can think of $P(M)$ as being a probability over models.
- This suggests we can work out the length of the encoded model as follows: $-\log P(M)$.
- Generally, we do not actually know $P(M)$: instead, we *compress* the model and measure the length of the compressed version.

The better we can compress our model, the more likely it will be.

Two-part coding

Putting these all together:

- Look for a model which assigns a high probability to the training set.
- . . . and one which can also be highly compressed.
- We can consider $-\log P(D \mid M)$ as being the length of a prefix encoding of the training set, in terms of the model.

MDL Principle

$$M^* = \text{argmin}_{m^*} l(P(D \mid M^*) + l(M^*)$$

This is a two-part code: one part for the data and one for the model itself.

Example

MDL for Language Modelling

Do we use a unigram or a bigram model for language modelling?

We need to compress both the set of unigrams and the set of bigrams and then apply the MDL Principle to tell us which one to select.

Example

- If we use a unigram model, then a simple compression scheme would be to assign each word a code word using the same w number of bits.
- There are N possible words.
- Ignoring counts, then the length of the model would be: $N \cdot w$ bits.
- (This ignores other details too).

Example

- If we instead use a bigram model (and assuming each bigram consists a word, followed by a special word and then the second word).
- Assuming we use w bits per word and there are N^2 possible bigrams
- We would use $N^2 \cdot 3w$ bits.
- The bigram model has a higher complexity than the unigram model.
- We say that bigram model has a corresponding lower probability than the probability of the unigram model.

Example

Which one we actually prefer also depends upon how well we model the data:

- if $I(P(D \mid M^u)) + I(M^u) < I(P(D \mid M^b)) + I(M^b)$ then we will prefer the unigram model M^u.
- Otherwise, we will prefer the bigram model M^b.

This is an objective way to rank possible models.

Example

- When there is a lot of training data, the term $I(P(D \mid M))$ will dominate.
- When there is a small amount if training data, we will be driven by the size of the compressed model.

MDL prefers simple models when there is little evidence otherwise.

Summary

- It is possible to think about learning as a search for simplicity.
- MDL shows how to use simplicity in learning.
- Compression can be used to assign probabilities to models.