L101: Machine Learning for Language Processing

Lecture 3

Guy Emerson

Today’s Lecture

- Discriminative Models
  - Logistic Regression
  - Maximum Entropy Markov Model
  - Conditional Random Field
- Named Entity Recognition

Recap – Models

- Generative – $P(x, y)$
- Discriminative – $P(y|x)$

Recap – Naive Bayes

$$\begin{aligned}\operatorname*{argmax}_y P(y|x) &= \operatorname*{argmax}_y P(y)\, P(x|y) \\ &\approx \operatorname*{argmax}_y P(y) \prod_i P(x_i|y)\end{aligned}$$

Discriminative – approximate $P(y|x)$?

Naive Bayes assumes that input features x_i are conditionally independent given the class y.

For a discriminative model, what kind of simplifying assumption can we make?

Logistic Regression

$$P(y|x) \approx \frac{1}{Z} \exp\left(\theta_y + \sum_i \theta_{y,i} x_i\right)$$

$$= \frac{\exp\left(\theta_y + \sum_i \theta_{y,i} x_i\right)}{\sum_{y'} \exp\left(\theta_{y'} + \sum_i \theta_{y',i} x_i\right)}$$

Logistic regression is also called a log-linear model, because the unnormalised log-probability of a class y is a linear function of the input features. The name “regression” refers to statistical regression (predicting the value of a continuous variable, based on input features), but here we are predicting log-probabilities. Taking the exponential makes each score positive. Normalising (using the normalisation constant Z) makes the scores sum to 1.

For comparison, we can write Naive Bayes in a similar normalised form:

$$P(y|x) = \frac{P(y) \prod_i P(x_i|y)}{P(x)}$$

$$Z = P(x) = \sum_{y'} \left(P(y') \prod_i P(x_i|y') \right)$$

Logistic Regression

$$P(y|x) \approx \frac{1}{Z} \exp\left(\theta_y + \sum_i \theta_{y,i} x_i\right)$$

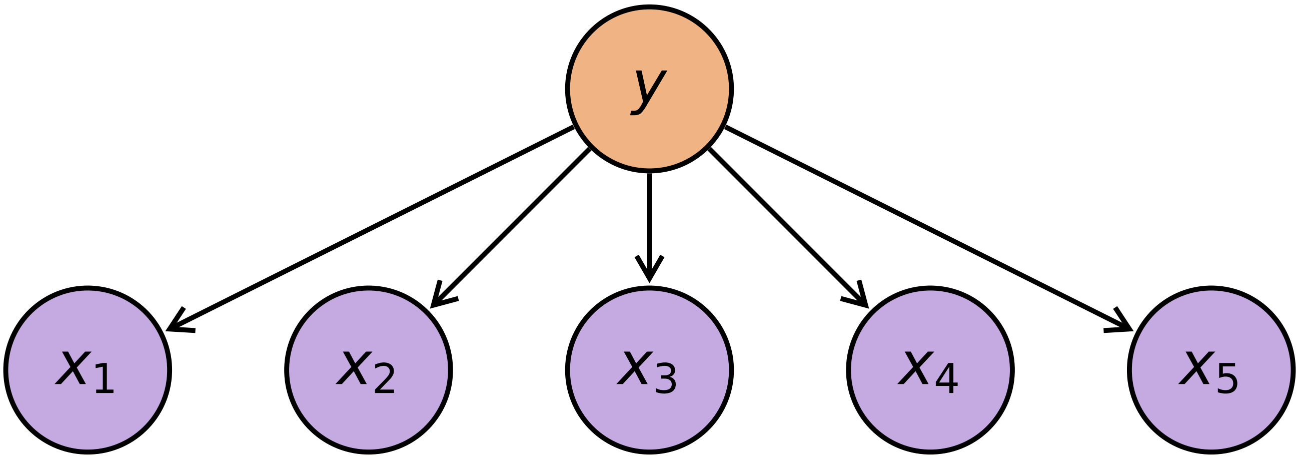
$$= \frac{\exp\left(\theta_y + \sum_i (\theta_{y,i} + k) x_i\right)}{\sum_{y'} \exp\left(\theta_{y'} + \sum_i (\theta_{y',i} + k) x_i\right)}$$

For one value of y, we can set $\theta_{y,i} = 0$ for all i. This is because, for any fixed i, adding a constant value k to each $\theta_{y,i}$ does not change the predictions of the model.

For example, if there are just two output classes, we only need one parameter for each input feature – this parameter says which output is more likely, given this feature.

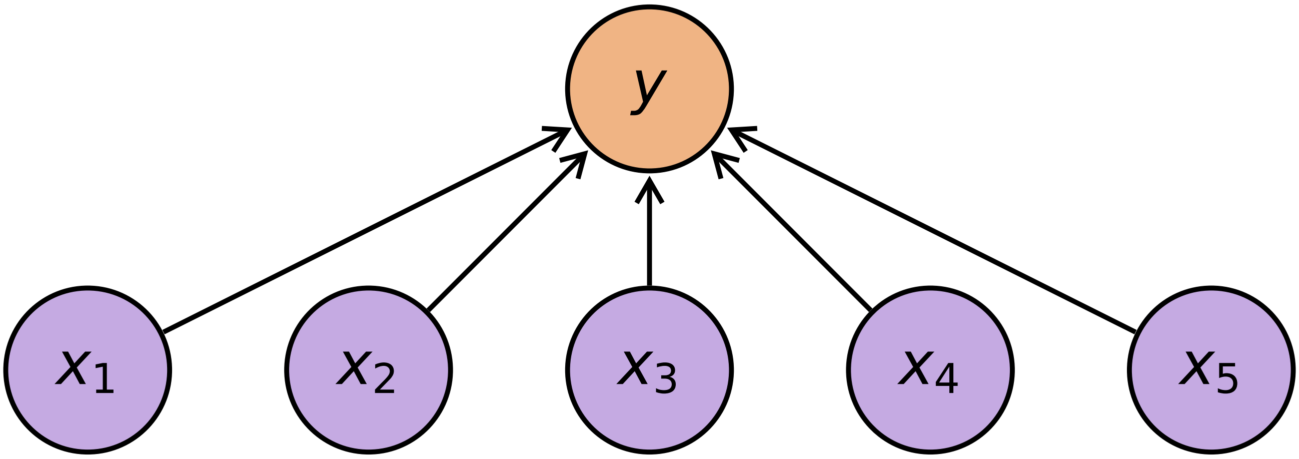
Logistic regression therefore has fewer parameters than Naive Bayes – precisely because we aren't modelling $P(x)$.

Naive Bayes



Naive Bayes as a probabilistic graphical model. Each feature x_i depends only on the class y.

Logistic Regression



Logistic regression has essentially the same structure, but with dependence in the opposite direction.

In principle, this diagram is much more general than logistic regression – in the following slides we will see how logistic regression is the simplest such model.

Logistic Regression

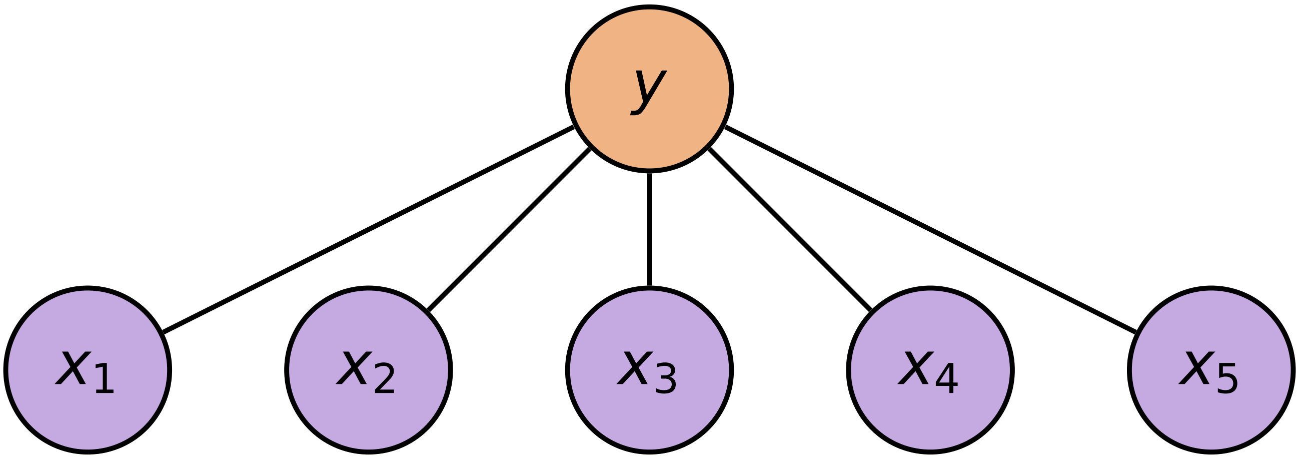
- Parameters: θ_y, $\theta_{y,i}$
- Optimise for: $\sum_{(x,y)\in D} \log P(y|x)$
- No closed form formula!

Unlike Naive Bayes, we're optimising $P(y|x)$, rather than $P(x, y)$.

Training is more difficult than for Naive Bayes (but still not too difficult).

Optimisation can be done using gradient descent. Other algorithms exist (e.g. see Steven Clark's lecture notes, for a discussion of Generalised Iterative Scaling).

Independence of Features



Logistic Regression and Naive Bayes have the same probabilistic independence structure.

(This can be formalised in terms of factor graphs – there is one factor between each feature and the class.)

Independence of Features

- Hong Kong vs. HongKong
- Naive Bayes:
  - $P(x_i|y)$ same
  - $P(y|x)$ over-estimated
- Logistic Regression:
  - $P(y|x)$ same
  - $P(x_i|y)$ never used!

Logistic Regression and Naive Bayes have the same probabilistic independence structure – but they parametrised differently and optimised differently. This is important when the independence structure is wrong.

Naive Bayes aims to fit $P(x, y)$, and so gets $P(y|x)$ wrong. Logistic regression aims to fit $P(y|x)$. In principle, we could train $P(x)$ alongside logistic regression, so that we can look at $P(x|y)$ – if we did this, we would see that $P(x|y)$ is wrong (e.g. undergenerating "Hong" and "Kong").

Why Log-Linear?

- Consider all distributions $P(y|x)$
- Under constraints:
  - $P(y|x_i)$ matches observed data
- Maximise conditional entropy $H(Y|X)$ on observed data

→ Logistic regression

Here we are considering binary features only! Entropy does not straightforwardly generalise to continuous variables (although some generalisations exist), and for discrete non-binary variables on a scale, entropy ignores the scale.

Intuition: To maximise entropy, we want as much independence as possible. The constraints only consider one feature at a time – this means we have independence of features, given the class. The maximum entropy model can then be written in the form of logistic regression. Now note that maximising $\log P(y|x)$ across the data fixes the constraints.

For non-binary features, we can simply use the same equation, even if it doesn't have the same justification.
Further reading – maintaining independence of features, but removing linearity, gives us "generalised additive models" (Hastie & Tibshirani, 2006) `https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781118445112.stat03141`

Regularisation

- Equivalent of smoothing
- Optimise objective function:

$$\mathcal{L} = \log P(y|x) - \lambda|\theta|$$

With Naive Bayes, a 0 count leads to 0 probabilities. With Logistic Regression, a 0 count leads to infinite parameter values. We want to avoid this!

L1 regularisation (also known as “lasso”) penalises the absolute value.

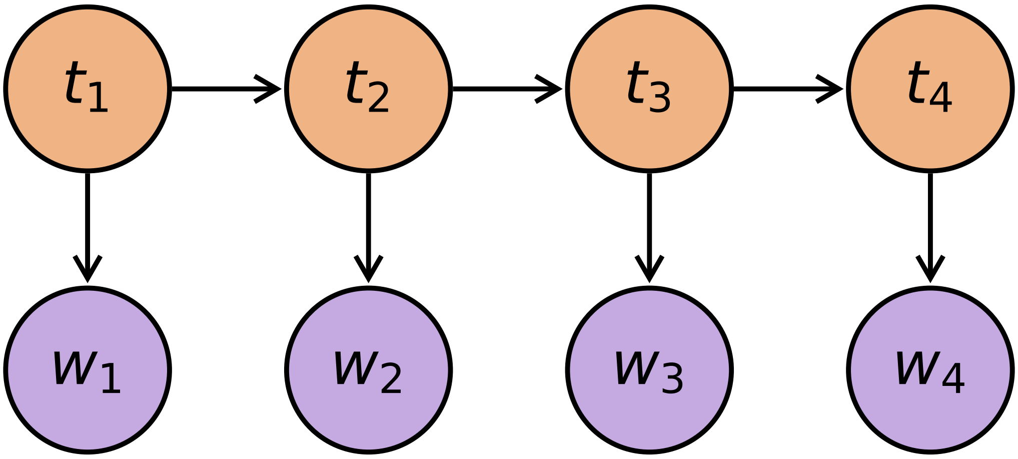
L2 regularisation (also known as “ridge”) penalises the square value.

Using both L1 and L2 regularisation (also known as “elastic”) is possible, and there are also other kinds.

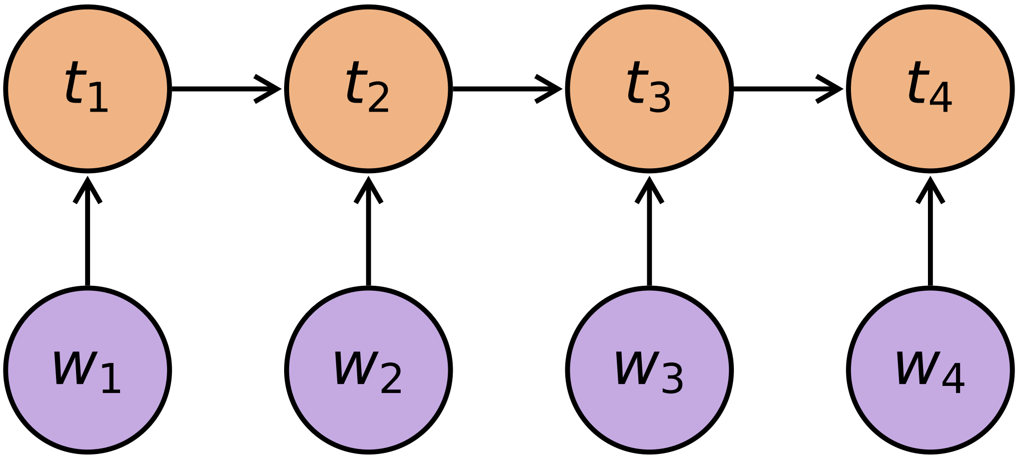
λ is a hyperparameter.

As well as avoiding parameters tending to infinity, regularisation also penalises parameters that are very large, which might indicate overfitting.

Recap: Hidden Markov Model



MaxEnt Markov Model



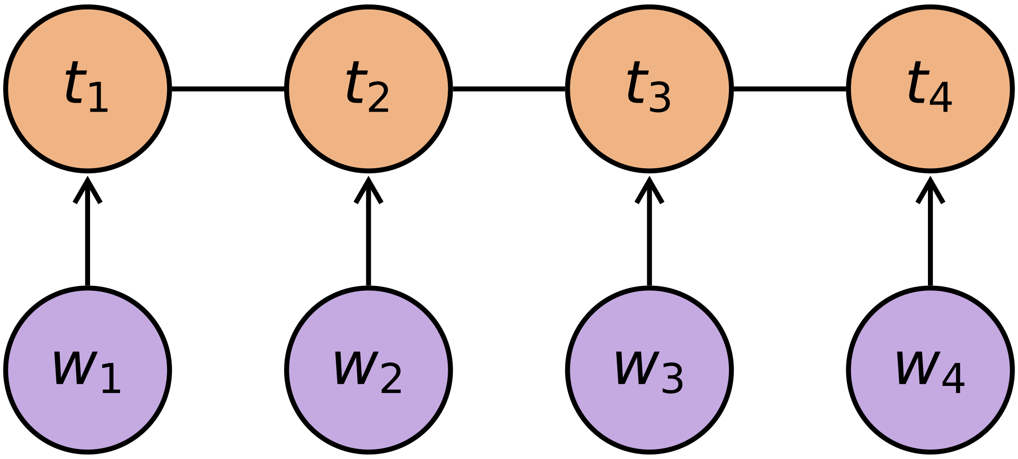
A Maximum Entropy Markov Model (MEMM) is to a Hidden Markov Model (HMM), just as Logistic Regression is to Naive Bayes – we have the same kind of structure, but parametrised differently, and optimised differently.

The difference between the two becomes important if we add more features that we know are not independent. With an HMM, it can be difficult to add features without risking breaking the predictions, but with an MEMM, it is much easier to add features.

MaxEnt Markov Model

- MaxEnt: logistic regression
- Markov: limited context
- Locally normalised: token by token
- Dynamic programming for inference

Conditional Random Field



Here we drop the assumption that each tag t_i is generated one at a time.

Conditional Random Field

- Conditional: discriminative
- Random field: undirected
- Globally normalised: all at once
- Dynamic programming or beam search for inference

The term “conditional random field” doesn’t include the Markov assumption. We can have both Markov and non-Markov conditional random fields – i.e. we can just look at n-grams (as depicted in the previous slide), or we can introduce dependence beyond n-grams.

If it’s Markov, we can still use dynamic programming. Otherwise, the non-local dependence means that we have to use an approximate algorithm like beam search.

Named Entity Recognition

Bill Gates says mosquitoes scare him more than sharks.

The reaction will produce 2,4- and 2,6-dinitrotoluene.

Named Entity Recognition

- Sequence labelling task
- Usually into classes: PER, LOC, etc.

Named Entity Recognition (NER) is important for many practical applications: search, information extraction, sentiment extraction...

It can also be useful as a preprocessing step before parsing, because names have different syntactic behaviour from other words.

The task usually assumes pretokenised input.

It is very domain- and genre-dependent – what counts as a named entity?

BIO scheme

Bill	Gates	says	mosquitoes
B-PER	I-PER	O	O

scare	him	more	than	sharks
O	O	O	O	O

B	beginning
I	inside
O	outside

Other schemes also exist – e.g. just I and O, or adding W for single-word names, or adding E for the end of a name.

The classes (such as PER above) also vary.

The tagging scheme matters a lot for performance.

Defining the Task

The New York Stock Exchange fell today.

Should “the” be included in the name? This needs to be consistently annotated.

Should “New York” be included as well as “New York Stock Exchange”? This would require a more sophisticated annotation scheme.

Defining the Task

The New York and Chicago
Stock Exchanges fell today.

Should we annotate “New York” and “Stock Exchange(s)” as being parts of one name? Again, this would require a more sophisticated annotation scheme.

Once annotation is decided, how should a system be evaluated? e.g. based on tokens or based on spans (sequences of tokens)?

Defining the Task

Queen Elizabeth
the Queen
the Queen of England
the queen of England
a queen of England
the queen of France

Which of these should count as a named entity?

Features for Named Entity Recognition

- Gazeteers (lists of names)
- Capitalisation
- Digits
- Punctuation
- Specific words preceding/following (Prof., Inc.)

This list is not exhaustive.

The features may not be independent, so we would generally prefer to use an MEMM, not an HMM.