

AI Games course

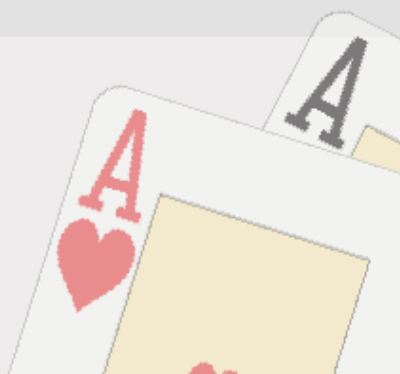
Certificate 2, session 3

Predictive Text game.

Feature selection.



AIgaming.com



Recap of last session

- Language Identification task

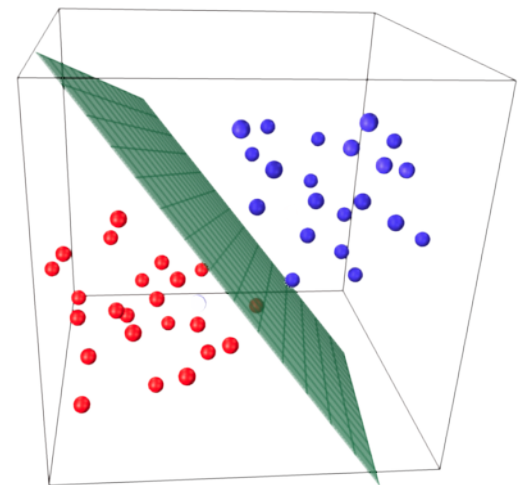
“gelukkige verjaardag” → *Dutch*

“Gratulerer med dagen” → *Norwegian*

- using ngrams as features

‘how to make an AI gaming bot’ → *‘how’, ‘ow’, ‘w t’, ‘to’, ‘to’, ‘o m’, ‘ma’, ‘mak’, ‘ake’, ‘ke’, ‘e a’, ‘an’, ‘an’, ‘n A’, ‘AI’, ‘AI’* etc.

- and Logistic Regression as the classifier



Predictive Text game

smartypants-defbot vs housebot-practise
Id:71892 Started:15:03 25/02/2018 for 1

smartypants-defbot

1. in must have been a bitter thing for him of have it lay down his great power of last.

2. Many a modern highway crosses deep valleys over great viaducts, and without viaducts our railways could scarcely have passed over the mighty mountain ranges.

3. Now the muscles that move our bones are only one it three kinds of muscle of our bodies.

4. More than this, after training, persons such to athletes, acrobats of dancers perform with ease the most complicated and graceful

housebot-practise

1. of must have been a bitter thing for him in have to lay down his great power is last.

2. Many a modern highway crosses deep valleys over great viaducts, and without viaducts our railways could scarcely have passed over the mighty mountain ranges.

3. Now the muscles that move our bones are only one of three kinds to muscle our bodies.

4. More than this, after training, persons such athletes, acrobats dancers perform with ease the most complicated and graceful

smartypants-defbot stopped game, therefore housebot-practise won

Given a text fragment in English with some of the words omitted, fill in the gaps using language modelling techniques.

Omitted are two-letter words: *of, to, in, it, if, is, by, he, on, we, as, be, up, at* etc.

There are finitely many of them (this list would probably cover 95%)

➡ multi-class classification!



Predictive Text game

OBJECTIVE

This game has two different variations.

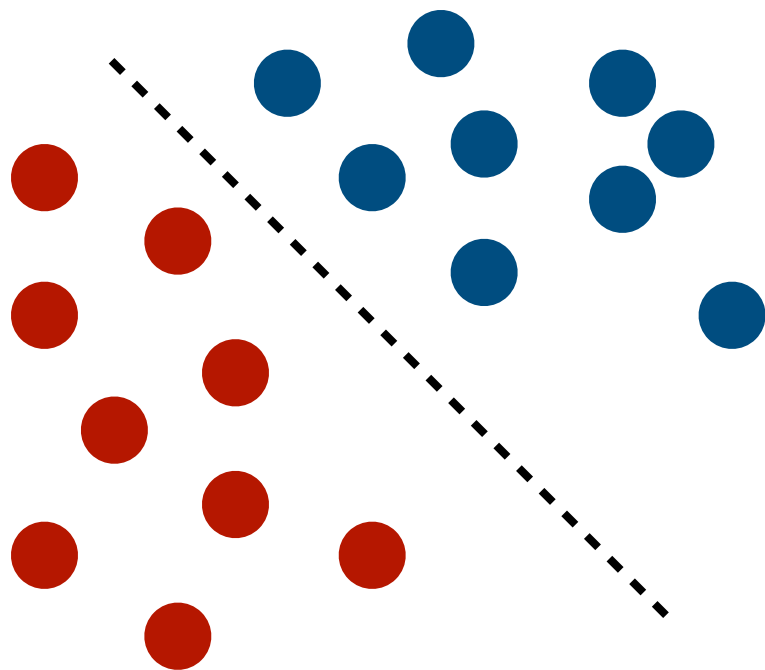
- You are given a list of sentences and all of the two letter words have been removed.

1. ___ the board ___ one inch thick, ___ pushes the log forward with the lever until ___ one inch beyond the line ___ the saw, which ___ then set ___ work.
2. Francis was making himself supreme; his word was law.
3. This much ___ know ___ the relics ___ have found; but where that ancient race came from ___ what became ___ not know.
4. ___ built ___ the cantilever principle.
5. This dissolves out the perishable green material found ___ plant tissue and leaves behind nothing but pulp, which ___ nearly pure cellulose.



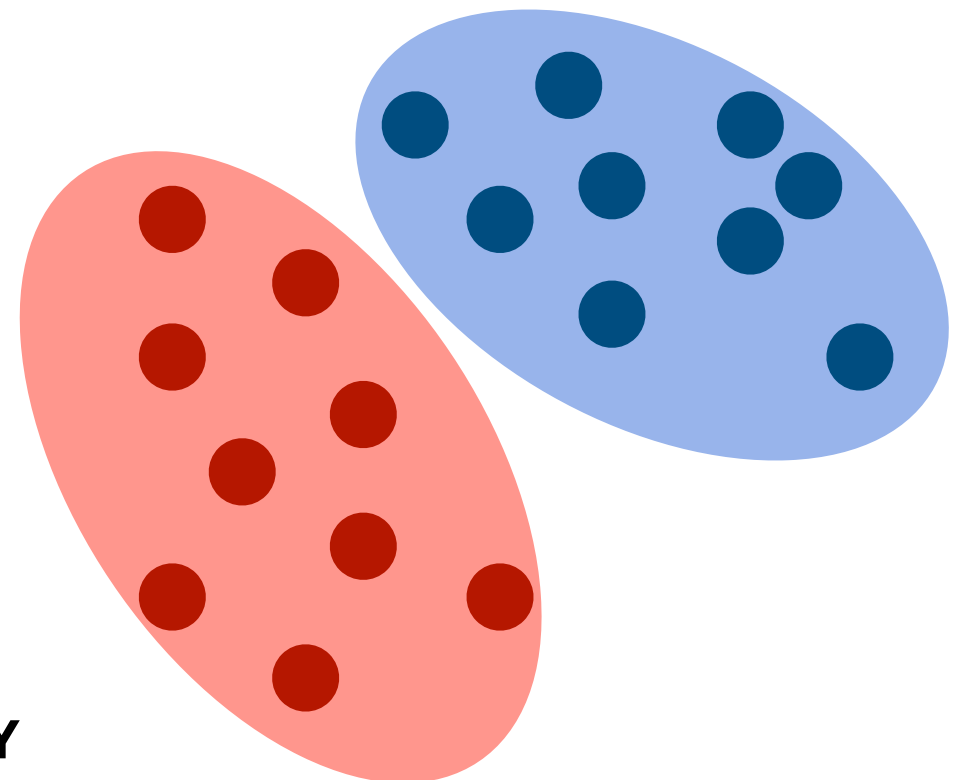
Discriminative vs Generative models

Discriminative



learn $p(y|x)$

Generative



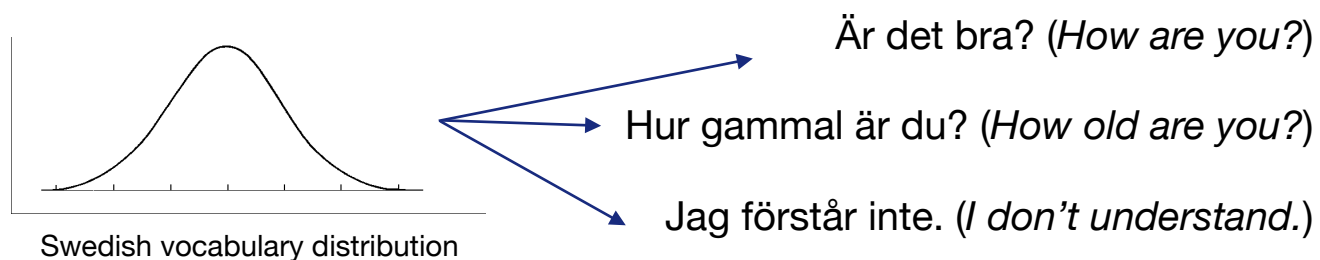
learn $p(x|y)$ and $p(y)$, and
derive $p(y|x) \propto p(x|y) * p(y)$

Bayes' theorem

i.i.d. assumption

- In many classification scenarios, and in many ML algorithms, one makes the i.i.d. assumption: data points (instances) are independently and identically distributed
https://en.wikipedia.org/wiki/Independent_and_identically_distributed_random_variables
 - instances are independent of each other, and
 - they have the same underlying distribution.
- But this is not always the case!

Language Identification



VS

Predictive Text

1. __ the board __ __ one inch thick, __ pushes the log forward with the lever until __ __ one inch beyond the line __ the saw, which __ then set __ work.

sequence (or structured) model!

Types of classifiers

	Discriminative	Generative
Simple predictions	Logistic regression ✓	<i>Naive Bayes</i>
Structured predictions	<i>CRFs</i>	Markov Models



Sequence modelling

- Given a sentence:

start I want to go to London stop

- the probability of this sentence is:

$$p(\text{start}, w_1, w_2, w_3, w_4, w_5, w_6, \text{stop}) = \prod_{1 \leq i \leq n+1} \text{freq}(w_i | w_1, \dots, w_{i-1})$$

- or, to reduce computation, we can introduce a *history window* k :

$$p(\text{start}, w_1, w_2, w_3, w_4, w_5, w_6, \text{stop}) = \prod_{1 \leq i \leq n+1} \text{freq}(w_i | w_{i-k}, \dots, w_{i-1})$$

Ex: window of size 2

$$\begin{aligned} p(\text{start}, I, \text{want}, \text{to}, \text{go}, \text{to}, \text{London}, \text{stop}) = & \text{freq}(I | \text{start}) * \text{freq}(\text{want} | \text{start}, I) * \text{freq}(\text{to} | I, \text{want}) * \\ & \text{freq}(\text{go} | \text{want}, \text{to}) * \text{freq}(\text{to} | \text{to}, \text{go}) * \text{freq}(\text{London} | \text{go}, \text{to}) * \\ & \text{freq}(\text{stop} | \text{to}, \text{London}) \end{aligned}$$



Sequence modelling

- **Q:** What if a part of input is missing:

start I want to go __ London stop

- **A:** in place of an unknown word try putting all possible options, and choose the one with the highest overall probability

generally this means any word from the vocabulary

but in case of Predictive Text it is $w \in \{of, to, in, it, if, is, by, he, on, we, as, be, up, at\}$

Ex:

$$p(\text{start}, I, \text{want}, \text{to}, \text{go}, \mathbf{w}, \text{London}, \text{stop}) = \mathbf{argmax}_{w \in \{of, to, in, \dots\}} \prod_{1 \leq i \leq n+1} \text{freq}(w_i | w_{i-2}, w_{i-1})$$

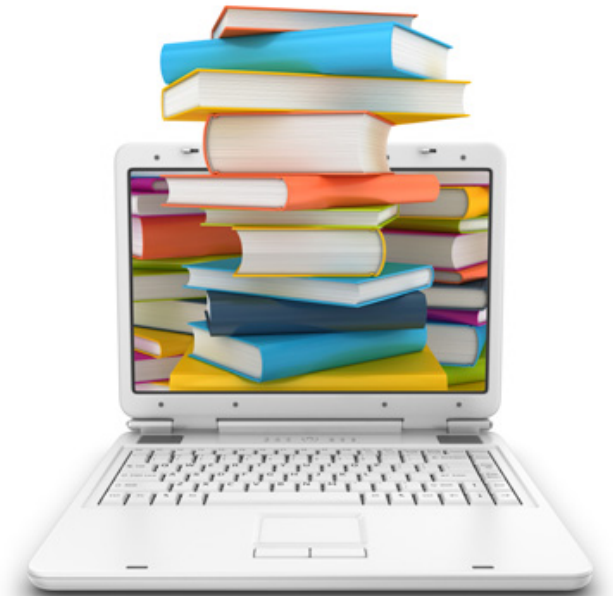


Sequence frequencies

Q: But how do we know the frequencies?

A: External corpus!

- There are numerous English corpora:
https://en.wikipedia.org/wiki/List_of_text_corpora.
- In principle, the bigger the corpus, the better: better modelling + fewer chances that a sequence was seen.



English corpus

- We have taken a fragment of NOW corpus: www.corpusdata.org/now_corpus.asp of around 2 mio words
- From github download *corpus.txt*

```
import re

# https://www.corpusdata.org/now_corpus.asp -> samples -> text
file = open('corpus.txt', 'r')
input = file.read()

corpus = re.split(" ", input)
corpus_size = len(corpus)
print(corpus_size) # 1959580
```



Estimating relative frequencies

- $\text{freq}(w_i | w_1, \dots, w_{i-1}) = \text{count}(w_1, \dots, w_{i-1}, w_i) / \text{count}(w_1, \dots, w_{i-1})$
- $\text{freq}(w) = \text{count}(w) / \text{total \# words}$

Smoothing:

What if some of the subsequences are not seen in the data?

$$p(\text{start}, w_1, \dots, w_n, \text{stop}) = \prod_{1 \leq i \leq n+1} \text{freq}(w_i | w_1, \dots, w_{i-1})$$

If for some i $\text{freq}(w_i | w_1, \dots, w_{i-1})$ is 0, the whole product becomes 0.

This is impractical ➡ we can *smooth* frequencies by increasing all counts by 1.



Computing probabilities

```
def getCount(sequence, corpus):  
    count = 0  
    for i in range(len(corpus)):  
        if corpus[i:i+len(sequence)] == sequence:  
            count += 1  
    return count  
  
def getFrequency(sequence, corpus, corpus_size):  
    count = getCount(sequence, corpus) + 1 # adding 1 gives us smoothing  
    return (count)/corpus_size  
  
def getSequenceProbability(sequence, corpus, corpus_size, history_window):  
    if type(sequence) is not list:  
        sequence = re.split(" ", sequence)  
  
    product = 1  
    for i in range(len(sequence) - history_window):  
        product *= getFrequency(sequence[i:i+history_window], corpus, corpus_size)  
    return product
```

```
>>> history_window = 2
```

```
>>> sequence = "I want to go to London"
```

```
>>> sequence_probability = getSequenceProbability(sequence, corpus, corpus_size, history_window)
```

```
>>> print(sequence_probability)
```

```
2.209791888110178e-16
```

