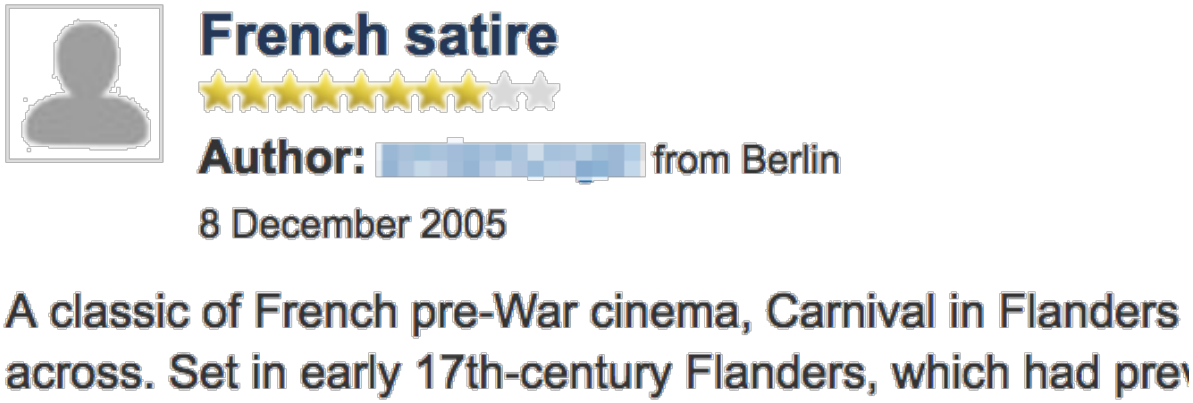
First text classification model

Sentiment classification

IMDB movie reviews dataset

- http://ai.stanford.edu/~amaas/data/sentiment/
- Contains 25000 positive and 25000 negative reviews



- Contains at most 30 reviews per movie
- At least 7 stars out of 10 → positive (label = 1)
- At most 4 stars out of 10 → negative (label = 0)
- 50/50 train/test split
- Evaluation: accuracy

Sentiment classification

Features: bag of 1-grams with TF-IDF values

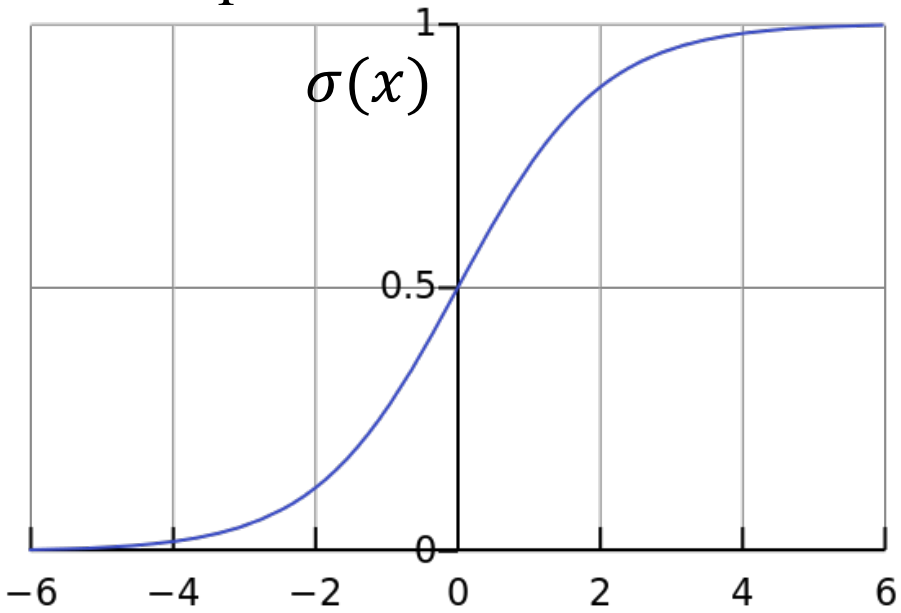
- 25000 rows, 74849 columns for training
- Extremely sparse feature matrix – 99.8% are zeros

acting	actingjob	actings	actingwise
0.000000	0.0	0.0	0.0
0.000000	0.0	0.0	0.0
0.053504	0.0	0.0	0.0
0.033293	0.0	0.0	0.0
0.000000	0.0	0.0	0.0

Sentiment classification

Model: Logistic regression

- $p(y = 1|x) = \sigma(w^T x)$
- Linear classification model
- Can handle sparse data
- Fast to train
- Weights can be interpreted



Sentiment classification

Logistic regression over bag of 1-grams with TF-IDF

- Accuracy on test set: 88.5%
- Let's look at learnt weights:

ngram	weight
great	9.042803
excellent	8.487379
perfect	6.907277
best	6.440972
wonderful	6.237365

Top positive

VS

ngram	weight
worst	-12.748257
awful	-9.150810
bad	-8.974974
waste	-8.944854
boring	-8.340877

Top negative

Better sentiment classification

Let's try to add 2-grams

- Throw away n-grams seen less than 5 times
- 25000 rows, 156821 columns for training

and am	and amanda	and amateur	and amateurish	and amazing
0.068255	0.0	0.0	0.0	0.0
0.000000	0.0	0.0	0.0	0.0
0.000000	0.0	0.0	0.0	0.0
0.000000	0.0	0.0	0.0	0.0
0.000000	0.0	0.0	0.0	0.0

Better sentiment classification

Logistic regression over bag of 1,2-grams with TF-IDF

- Accuracy on test set: 89.9% (+1.5%)
- Let's look at learnt weights:

well worth	13.788515		bad	-24.467648
best	13.633200		poor	-24.319746
rare	13.570259	VS	the worst	-23.773352
better than	13.500025		waste	-22.880340
Near top positive			Near top negative	

How to make it even better

Play around with tokenization

- Special tokens like emoji, ":)" and "!!!" can help

Try to normalize tokens

- Adding stemming or lemmatization

Try different models

- SVM, Naïve Bayes, …

Throw BOW away and use Deep Learning

- https://arxiv.org/pdf/1512.08183.pdf
- Accuracy on test set in 2016: 92.14% (+2.5%)

Summary

- Bag of words and simple linear models actually work for texts
- The accuracy gain from deep learning models is not mind blowing for sentiment classification
- In the next video we'll look at spam filtering task