Machine learning with many, many labels

Dr. Mike Swarbrick Jones

ODSC Workshop September 2018

Plan for workshop

- Introduce problem
- Primer in text classification (on a new dataset)
- Why you want to avoid this problem
- *Intermission*
- Talk in depth about ML problems/solutions when you have a lot of classes

Please do not wait until the end to ask questions

Classification

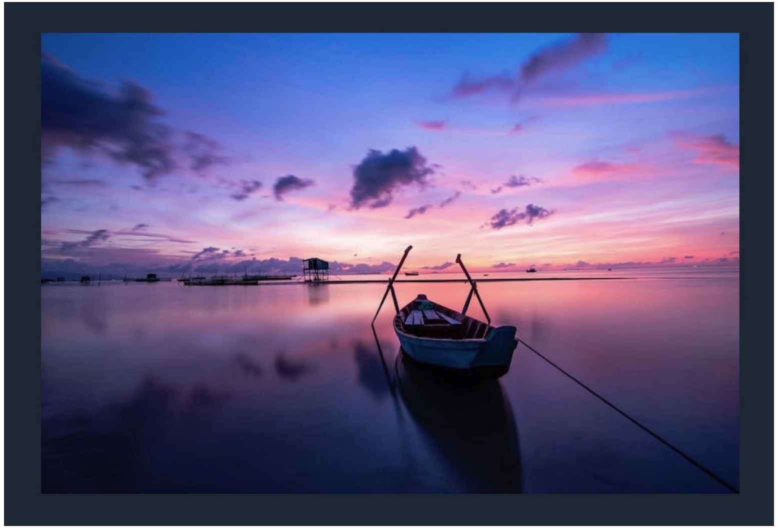
Given a piece of data, try and identify its category

Useful examples :

- Tag emails as spam and non-spam
- Diagnose an illness based on symptoms
- Classify road signs from images

Classification vs. labelling

Multi-labelling is when your data can have more than one class (not this talk)



PREDICTED CONCEPT	PROBABILITY
sunset	0.997
water	0.995
dawn	0.986
dusk	0.982
boat	0.981
reflection	0.977
evening	0.976

EvolutionAI

How many classes?

< 100 classes : big woop...

100s - 10,000s classes : focus of this talk

100,000 classes + : ‘extreme classification’ (XML)

Example : Image classification

Example : Image classification

ILSVRC (ImageNet)

Year of release : 2010

~ 1.4 million images

1000 classes

~ 1400 examples per class

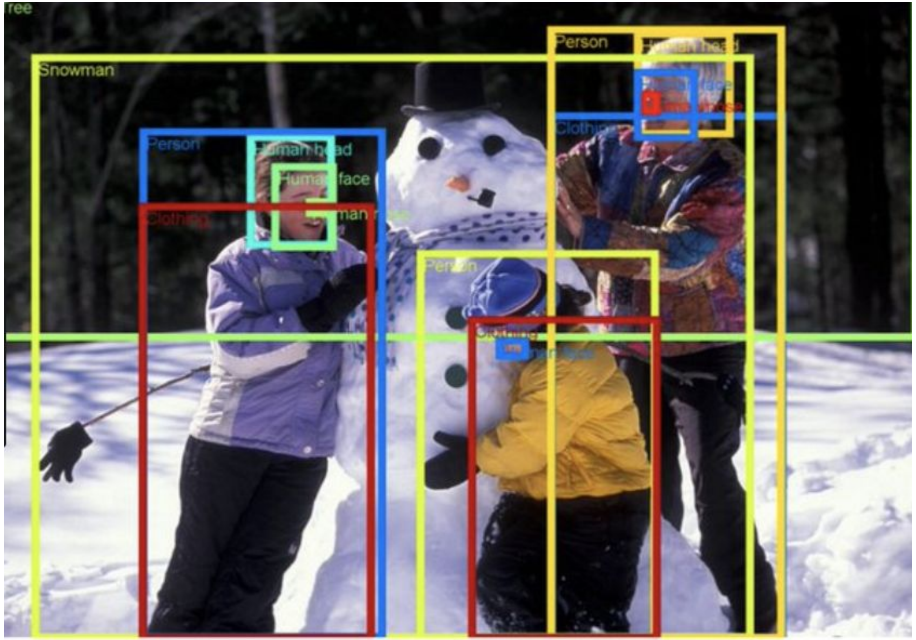
Image classification

JFT-300m (Google internal [JFT15])

Year of release : 2015

300M images

> 18,000 classes (multilabel)



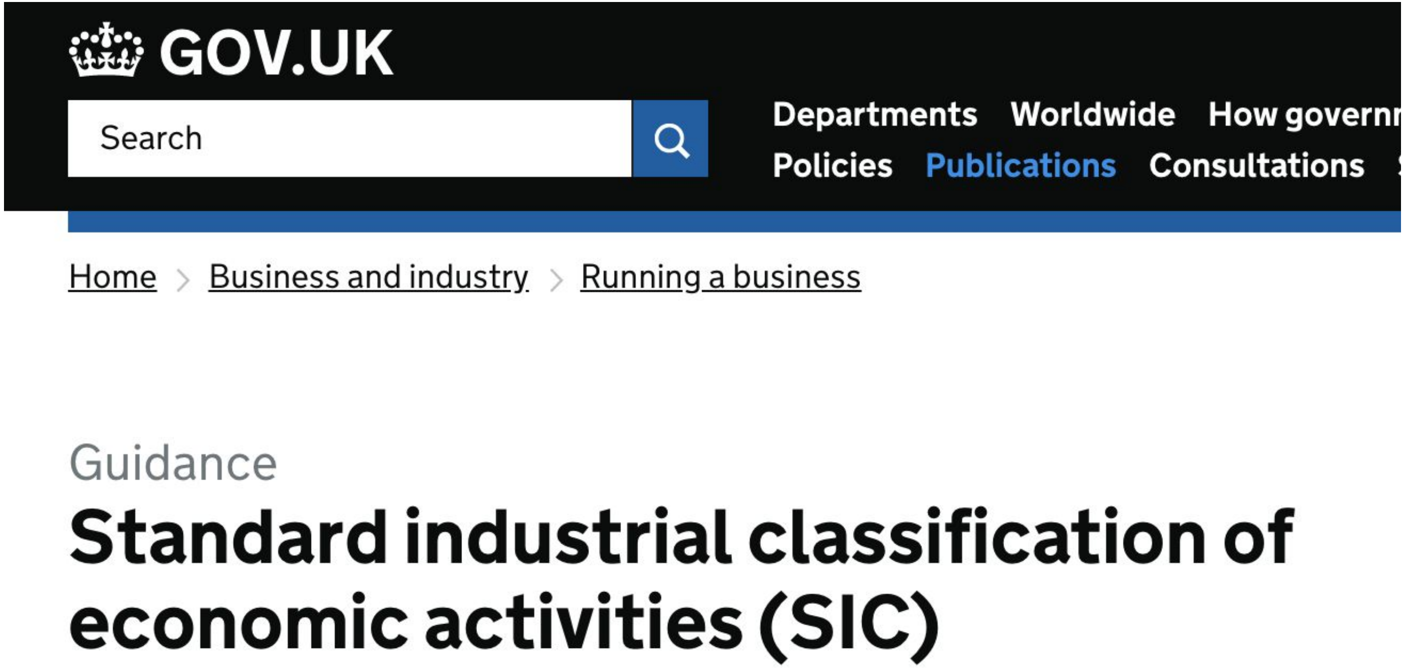
ImageNet break!

Language modelling

I took my _____ for a walk, she wasn't happy

Given a set of words, guess the missing word. Vocabularies are generally huge, can be hundreds of thousands.

SIC codes


GOV.UK
Search
Departments Worldwide How governr
Policies Publications Consultations
Home > Business and industry > Running a business
Guidance
Standard industrial classification of economic activities (SIC)

SIC codes

24410 - Precious metals production
62011 - Ready-made interactive leisure and entertainment software development
25940 - Manufacture of fasteners and screw machine products
10500 - Manufacture of dairy products
46330 - Wholesale of dairy products, eggs and edible oils and fats
81210 - General cleaning of buildings
20301 - Manufacture of paints, varnishes and similar coatings, mastics and sealants
64110 - Central banking
82920 - Packaging activities
94120 - Activities of professional membership organisations

SIC codes

Useful for

- understanding the economy
- understanding business clients (Know Your Customer)

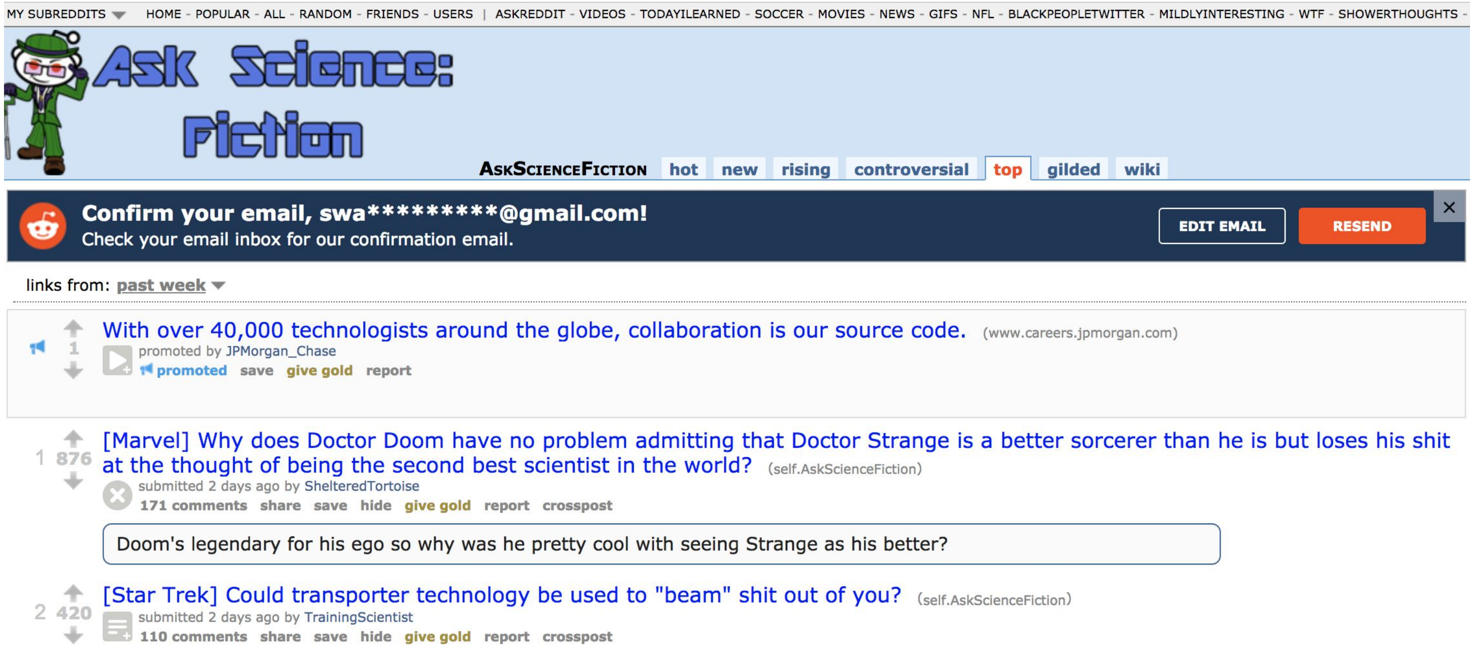
There are between 300 and 15,000 SIC codes depending on who you are talking to.

Suggestions from the audience?

Reddit Self-Posts (new!)

Goal was to find a text dataset which had many classes + many examples per class

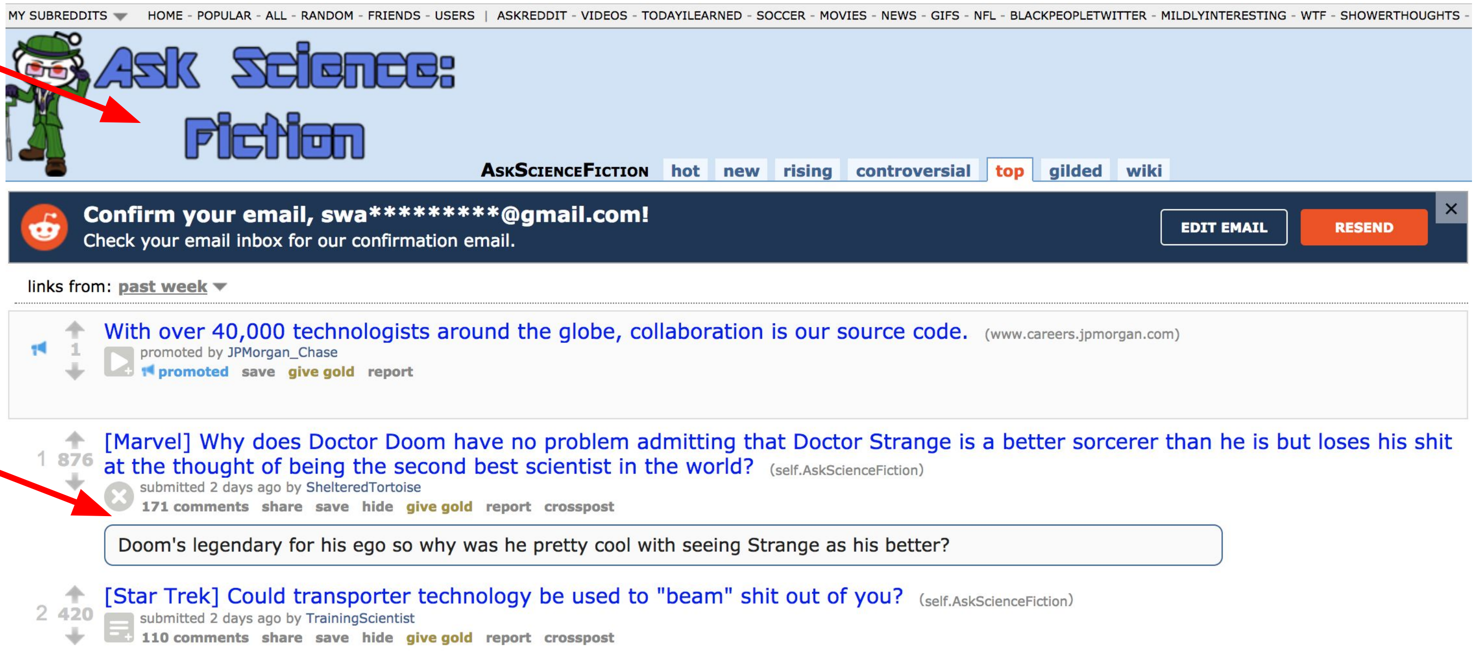
Reddit Self-Posts (new!)



Reddit Self-Posts (new!)

subreddit

selfpost



Reddit Self-Posts (new!)

Downloaded all self-posts from the last 2 years

Took all subreddits with at least 1000 posts (about 3000)

Manually classified subreddits into different topics (> 1000)

Topics needed to be very specific and unique (not stuff like r/askreddit)

GOTO NOTEBOOK

Primer in text classification

We're going to be talking about 'bag-of-words' classification

(other text classification algorithms are available)

Bag of words model

We start with ‘documents’

```
the cat sat on the mat
```

```
the dog sat on the cat
```

Bag of words model

Simply count the number of occurrences of each word

`the cat sat on the mat`

```
{`the` : 2, `cat` : 1, `sat`: 1, `on`: 1, `mat` : 1}
```

`the dog sat on the cat`

```
{`the` : 2, `cat` : 1, `sat`: 1, `on`: 1, `dog` : 1}
```

Bag of words model

Convert this to an array of word counts

```
the cat sat on the mat
```

```
{`the` : 2, `cat` : 1,
 `sat`: 1, `on`: 1,
 `mat` : 1}
```

$$
\begin{array}{cc}
\begin{pmatrix} 2 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix} &
\begin{matrix} \texttt{the} \\ \texttt{cat} \\ \texttt{sat} \\ \texttt{on} \\ \texttt{mat} \\ \texttt{dog} \end{matrix}
\end{array}
$$

Bigrams

So far : ‘dog bites man’ and ‘man bites dog’ will map to the same array

To encode some word ordering we can also look at pairs of words :

man bites dog -> {‘man’ : 1, ‘bites’ : 1, ‘dog’ : 1, ‘man bites’ : 1, ‘bites dog’ : 1}

Aside : when to use bag of words?

Google's recommendation [G18] -

"From our experiments, we have observed that the ratio of "number of samples" to "number of words per sample" correlates with which model performs well.

When the value for this ratio is < 1500 ... [bag of words models are good choice]"

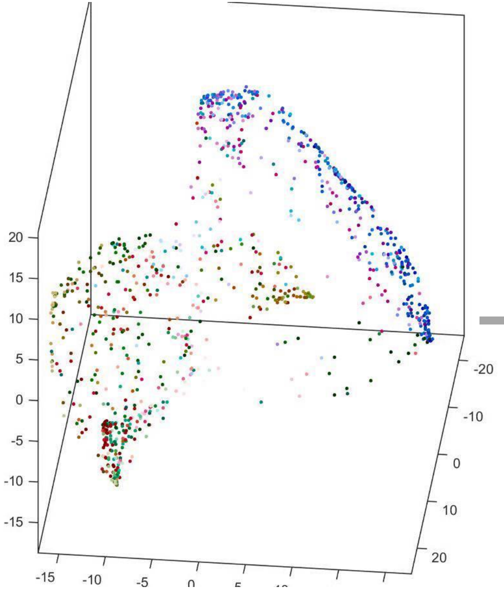
Next step : machine learning!

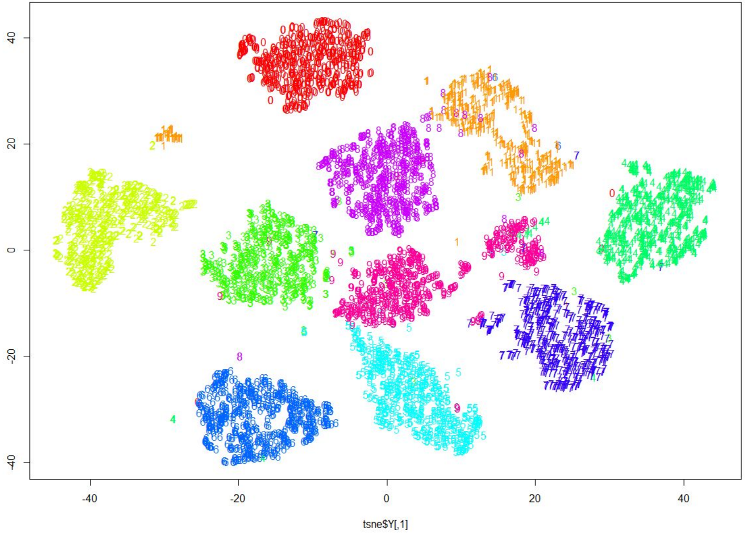
GOTO NOTEBOOK

t-SNE

t-SNE is a cool way to visualise high dimensional data in a 2d plot.

t-SNE





GOTO NOTEBOOK

Reasons you should use t-SNE

Bad reasons

- Anything useful

Good reasons

- Very fun
- Managers love it

Why is lots of labels hard? (for humans)

Good, large taxonomies are hard to make

You want your taxonomy to be granular enough to be useful

Not so granular that classification is impossible

Advice - build a hierarchy, breadth first

Overlapping classes are inevitable

E.g. from reddit dataset: r/astronomy and r/telescopes - often talking about the same things - should they be separate?

Not easy for humans

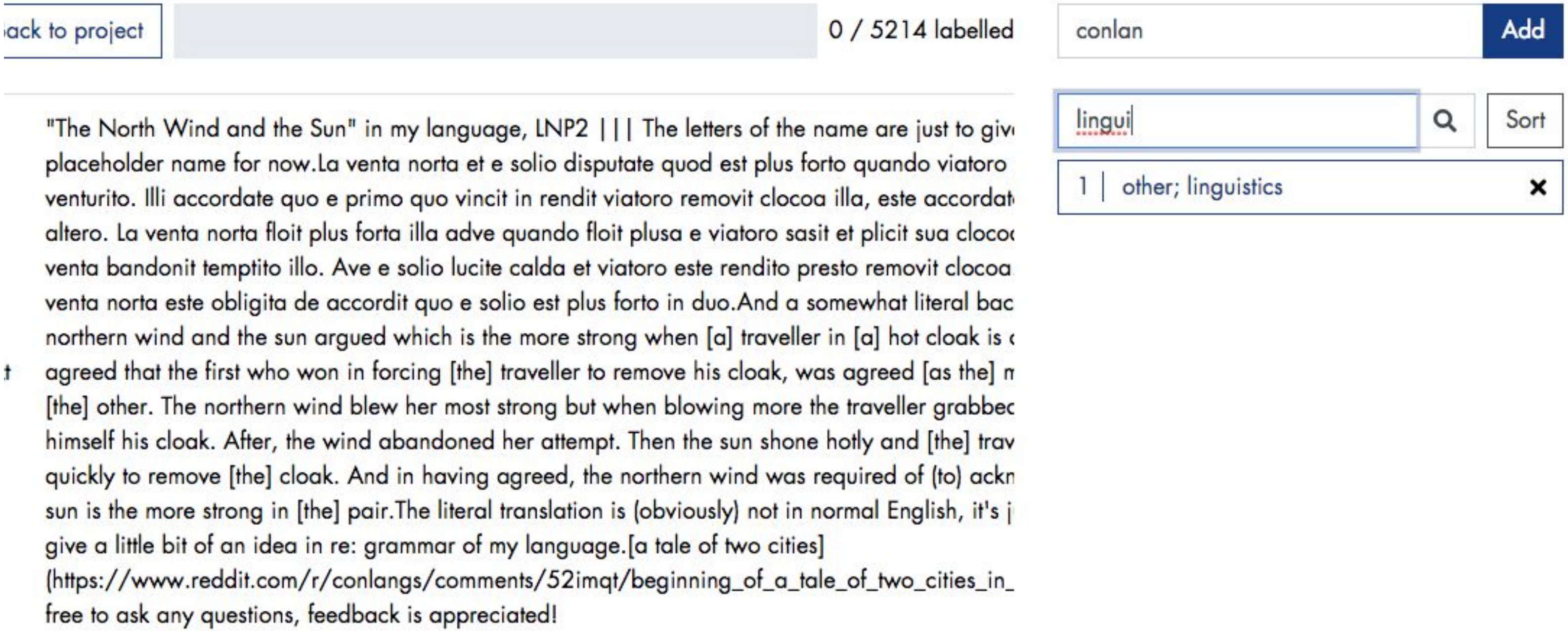
We are not good at remembering so many classes.

Human performance on ImageNet is about 20% error rate without training, about 3% with training (best ML model to date is about 2.4%)

Hard and expensive to get a high quality dataset

Labelling interfaces need thought

Labelling interfaces need thought



The big question : do you really need it?

Sometimes the only winning move is not to play!

Do you *need* high granularity?

Even if you managed to get a model that can accurately classify 10,000 classes… *so what?*

Do you really have 10,000 different actions you need to take based on this prediction? Can you lump classes together based on the business use-case?

Do you *need* high granularity?

If you have a hierarchy of classes - can you use higher levels?

Do you *need* high granularity?

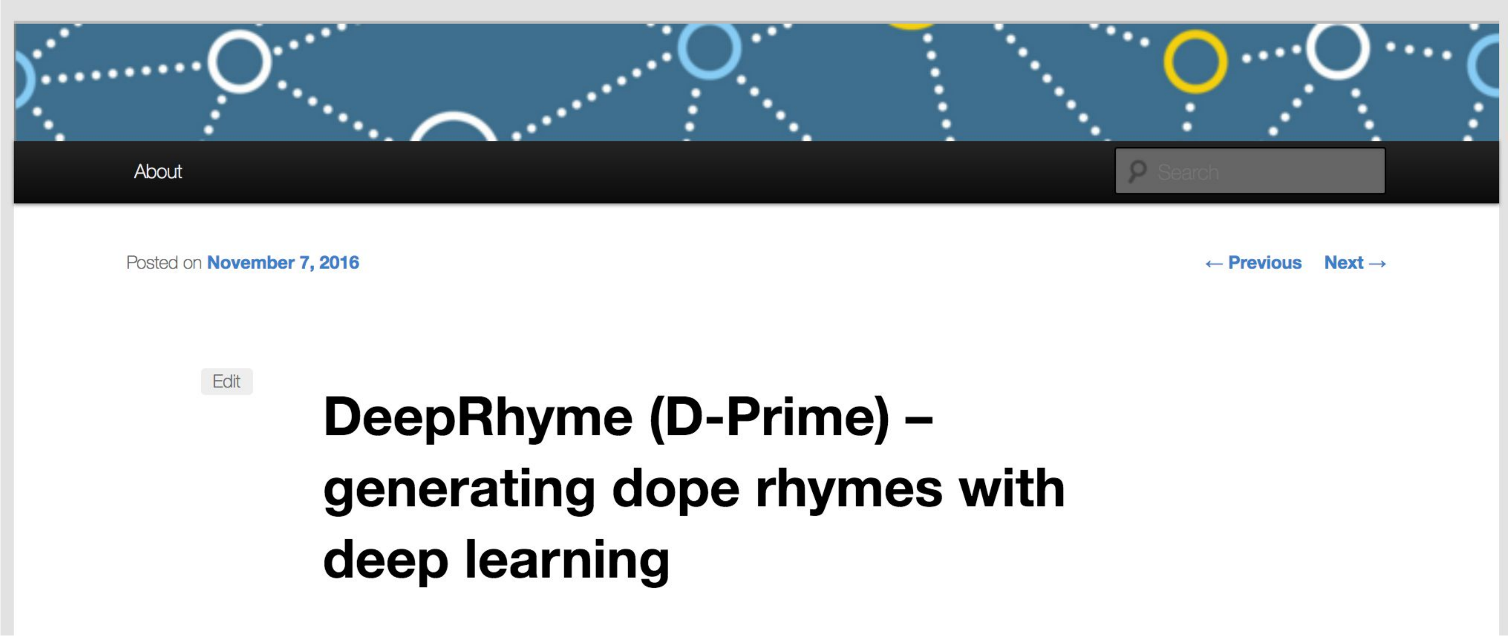
When you split a class into subclasses

- You make the problem more difficult (for humans and machines)
- You lower the average amount of data per class you will have

Is a subset of the classes good enough?

Can you get away with only being able to accurately predict the most important classes?

Is a subset of the classes good enough? [SJ16]



Is a subset of the classes sufficient?

DeepRhyme only learned to classify the most common few thousand words - it was enough!

```
don't want no money , i gotta make a mil
everytime you see me in the back of my grill
i'm just tryin to take a look at this ho
but when i hit the flo , we make it glow
yo , i remember when we used to do shows
been around the world and that's just how it goes
```

Intermission

GOTO MUFFINS

Why is this problem hard (for machines)?

Class imbalance

Typically you find that the most common classes are orders of magnitude more frequent than the least common classes.

Leads to general problems associated with ‘class imbalance’.

What metric do you want?

With class imbalance, you need to think hard about what metric you are using.

E.g. if 90% of all examples belong to one class, can get 90% prediction accuracy (precision) by always predicting the majority class!

Macro metrics

Maybe more appropriate : ‘macro’ metrics.

E.g. macro-precision : look at accuracy on each class label individually, then average across all classes.

This means your model needs to be good on both common labels and rare ones to get a good score.

Other metrics are available...

$$\mathrm{DCG@}k := \sum_{l \in \mathrm{rank}_k(\hat{\mathbf{y}})} \frac{\mathbf{y}_l}{\log(l+1)}$$

$$\mathrm{PSDCG@}k := \sum_{l \in \mathrm{rank}_k(\hat{\mathbf{y}})} \frac{\mathbf{y}_l}{p_l \log(l+1)}$$

$$\mathrm{nDCG@}k := \frac{\mathrm{DCG@}k}{\sum_{l=1}^{\min(k, \|\mathbf{y}\|_0)} \frac{1}{\log(l+1)}}$$

$$\mathrm{PSnDCG@}k := \frac{\mathrm{PSDCG@}k}{\sum_{l=1}^{k} \frac{1}{\log(l+1)}}$$

What metric do you want?

Really depends on the application - think it through before you start building models, make sure it aligns with your business use-case.

GOTO NOTEBOOK

Feature selection

Rather than use all features (words / n-grams) let's only use the useful ones (less is sometimes more).

When your data is imbalanced, need to be careful, to avoid bias towards the biggest classes

Feature selection

E.g. when doing text classification, DO NOT just take the most common words

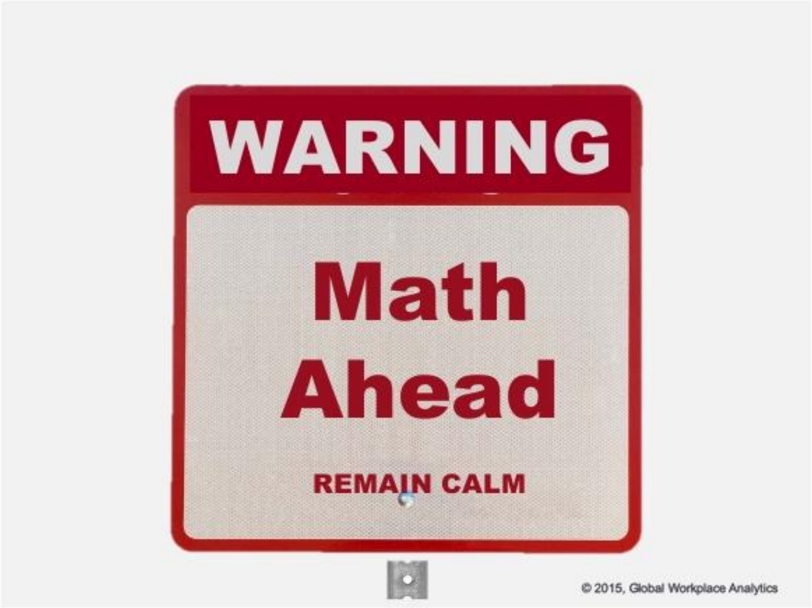
GOTO NOTEBOOK

Feature selection

Correlation coefficient is also biased towards most common classes.

For more robust ideas, recommend researching into imbalanced data feature selection (e.g. [YGXWQ13])

Model complexity necessarily grows



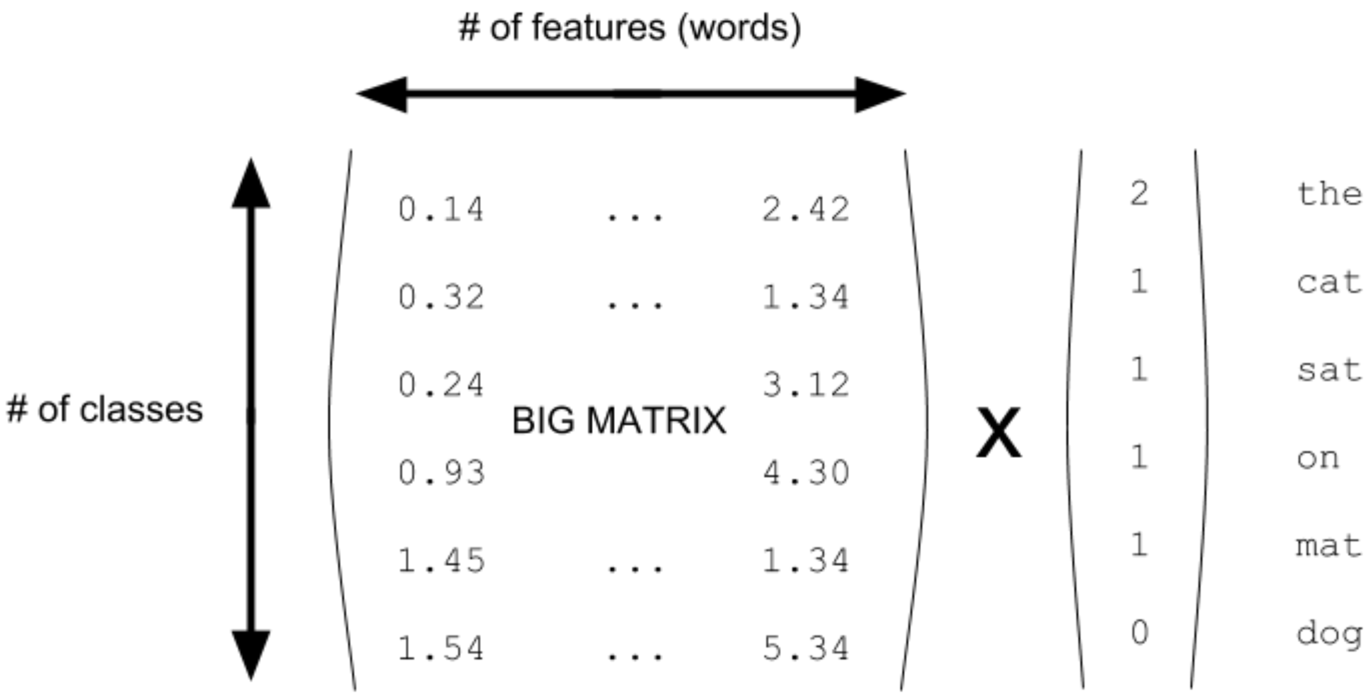
Model complexity necessarily grows

At the heart of most bag-of-words models (Naive Bayes / Logistic Regression etc), is a big matrix multiplication.

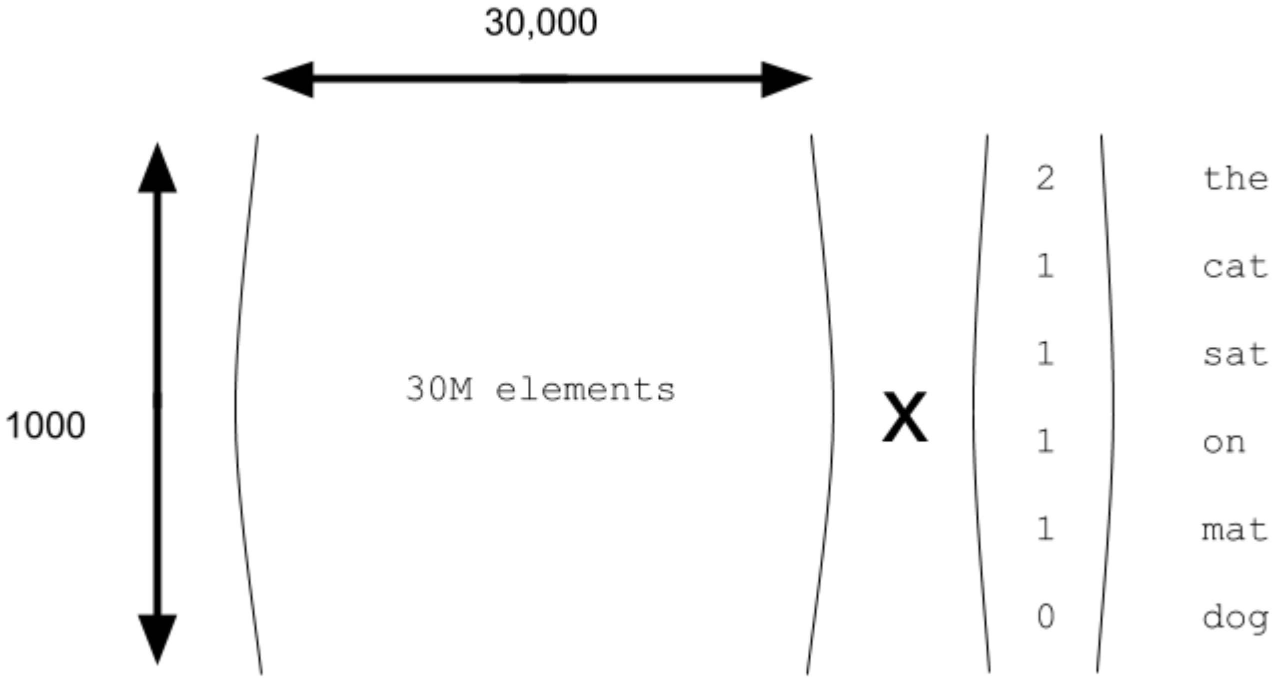
Matrix multiplication

$$\begin{pmatrix} 0.14 & \dots & 2.42 \\ 0.32 & \dots & 1.34 \\ 0.24 & & 3.12 \\ & \text{BIG MATRIX} & \\ 0.93 & & 4.30 \\ 1.45 & \dots & 1.34 \\ 1.54 & \dots & 5.34 \end{pmatrix} \times \begin{array}{cc} \left(\begin{array}{c} 2 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{array}\right) & \begin{array}{l} \texttt{the} \\ \texttt{cat} \\ \texttt{sat} \\ \texttt{on} \\ \texttt{mat} \\ \texttt{dog} \end{array} \end{array}$$

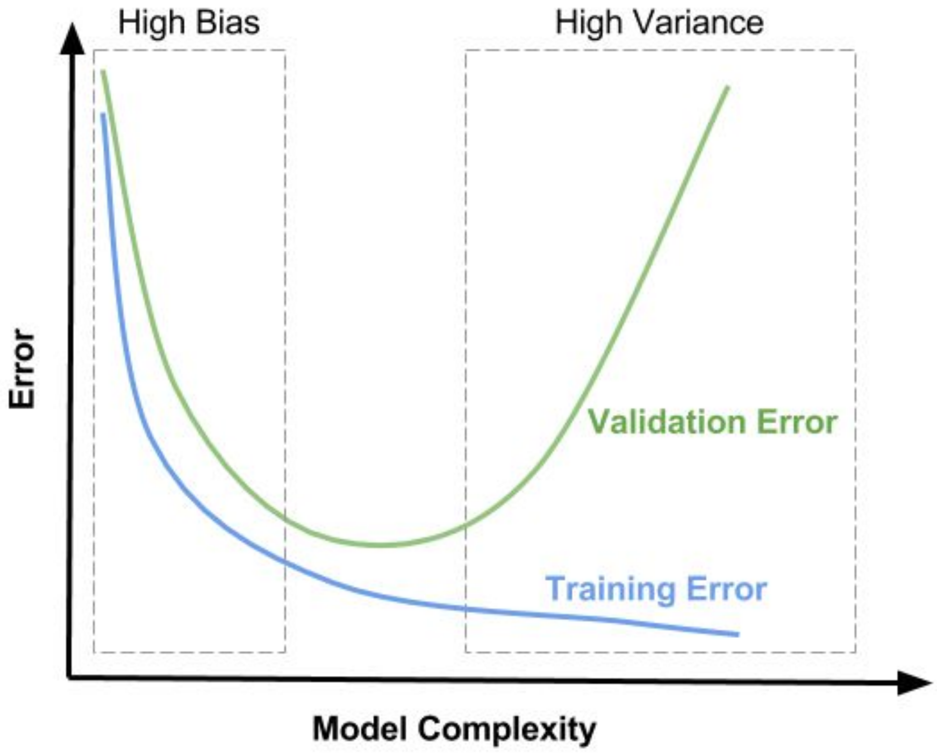
Matrix multiplication



Matrix multiplication



The bias variance trade-off



Model complexity is bad

As a rough rule of thumb, if the number of parameters you have is much greater than the number of training examples - bad things are going to happen to you...

Word embeddings

Representing each word as a learned vector, length 64 (say)

Word embeddings

Representing each word as a learned vector, length 64 (say)

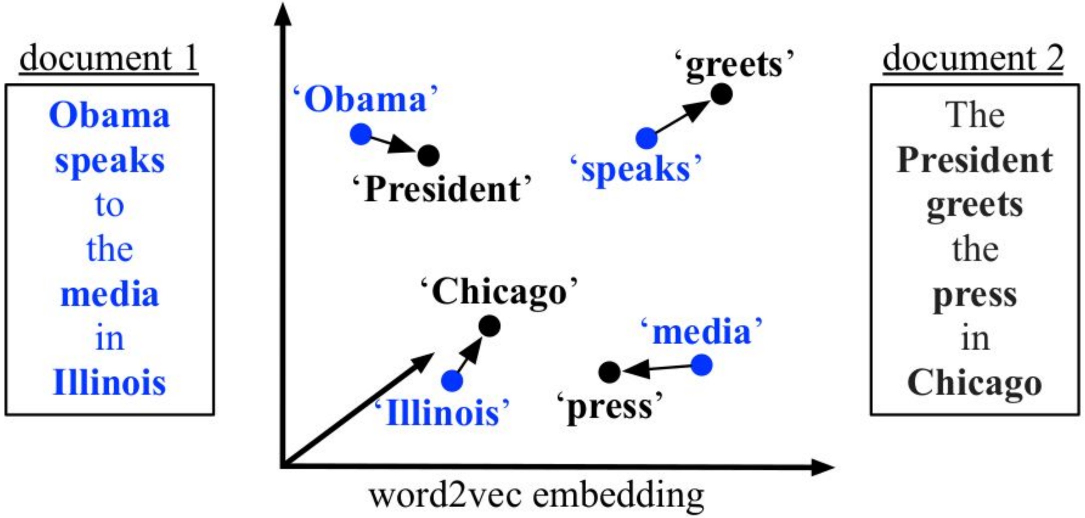
```
the cat sat on the mat
```

→

-0.1,-0.3, 0.4,…	the
0.3,-0.4, 0.5,…	cat
0.1, 1.3, 0.4,…	sat
0.5,-1.2, 0.9,…	on
-0.1,-0.3, 0.4,…	the
0.1,-0.2, 1.3,…	mat

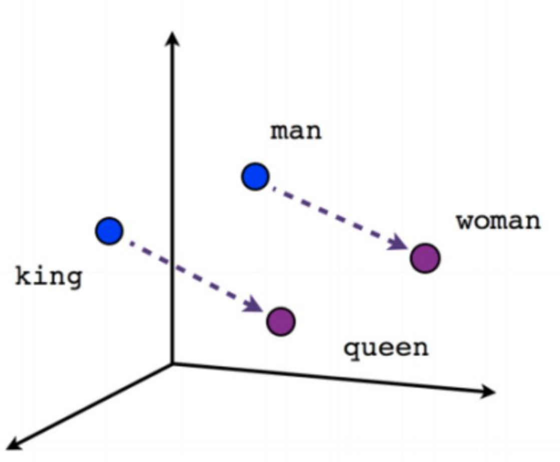
Word embeddings

When you learn word embeddings, typically they are positioned spatially based on meaning

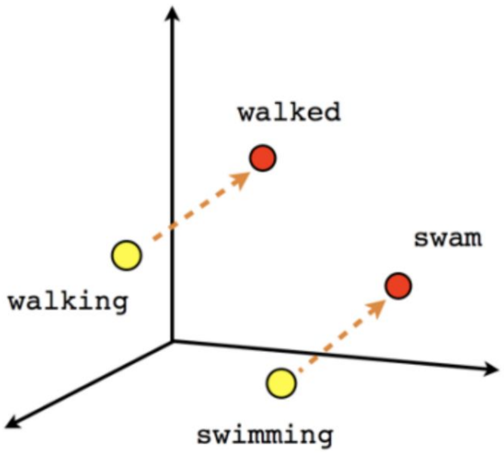


Word embeddings

More surprisingly, *relations* between words sometimes get encoded



Male-Female



Verb tense

Averaged word embeddings

Idea : represent each document by the average of the word embeddings

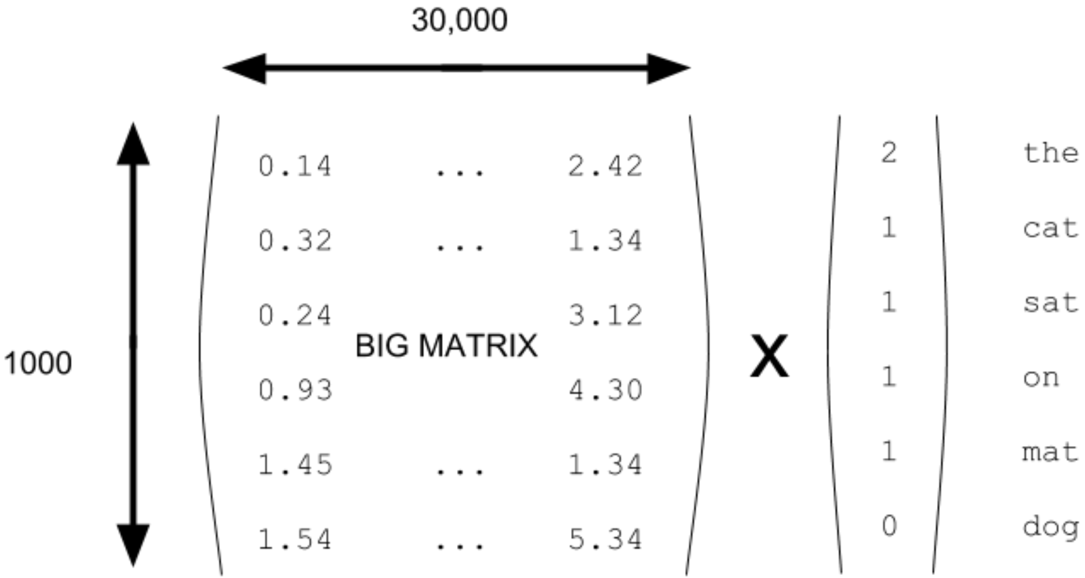
the cat sat on the mat →

-0.1,-0.3, 0.4,…	the
0.3,-0.4, 0.5,…	cat
0.1, 1.3, 0.4,…	sat
0.5,-1.2, 0.9,…	on
-0.1,-0.3, 0.4,…	the
0.1,-0.2, 1.3,…	mat

average: **0.1,-0.2, 0.6**

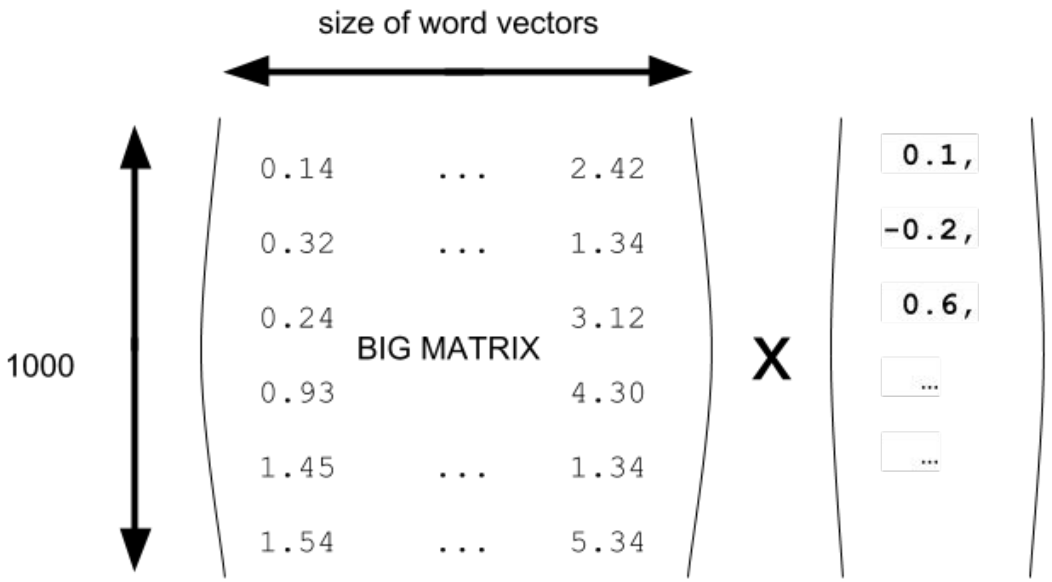
Matrix multiplication

Idea : use this as the document vector



Matrix multiplication

Idea : use this as the document vector



FastText (vanilla)

Instead of learning the matrices and embeddings directly, learn them with gradient descent. This is a simplified version of 'FastText' by FAIR [JGBM16]

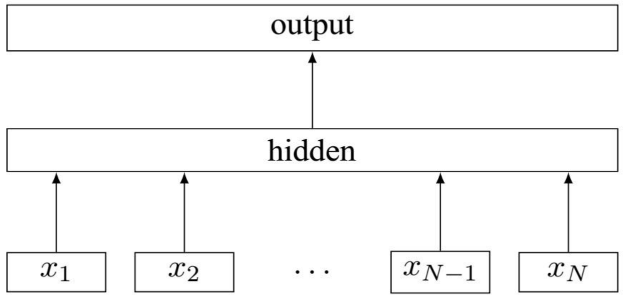


Figure 1: Model architecture of `fastText` for a sentence with N ngram features $x_1, \ldots, x_N$. The features are embedded and averaged to form the hidden variable.

FastText (vanilla)

FastText is a better choice for bag-of-words model with lots of classes

- Has fewer learned parameters
- Allows the model to share word representations between classes
- It is fast!

Aside : matrix decomposition

word embeddings

$$\left(1300 \times 30000 \right) = \left(1300 \times 64 \right) \times \left(64 \times 30000 \right)$$

GOTO NOTEBOOK

Look for linear algebra hacks like this

When you have a lot of classes -linear algebra tricks like this are everywhere - be on the lookout!

Dataset shift

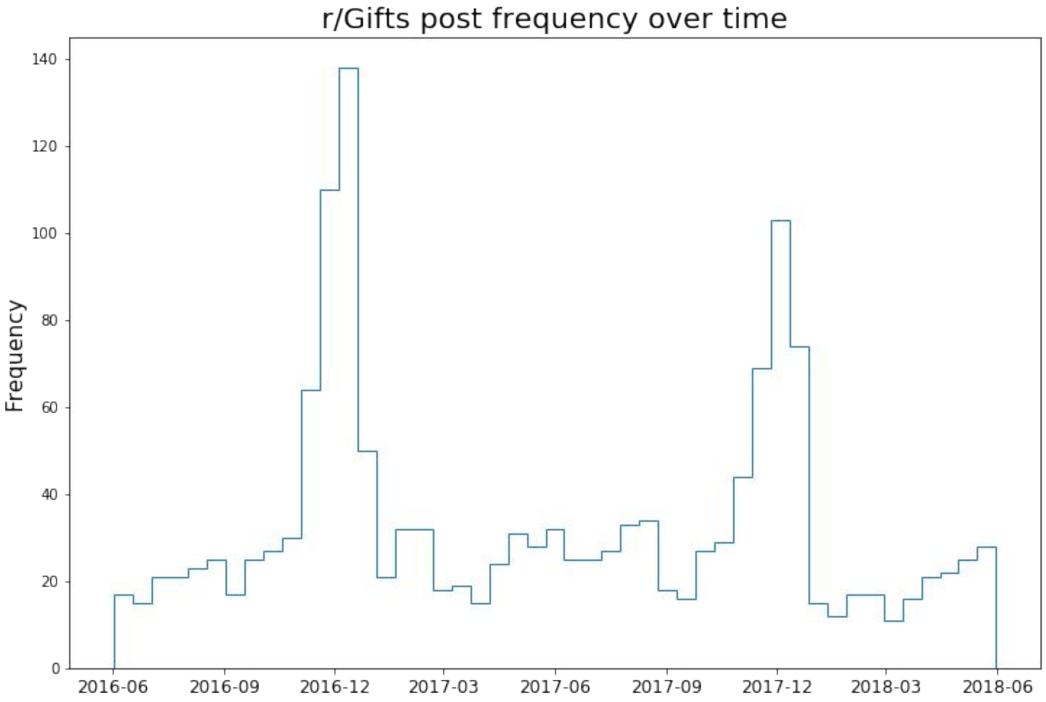
When we do supervised machine learning we want to assume that our training data is representative of what we're eventually going to run the model on (new, unseen data)

Bad assumption

This is generally not the case when you have a lot of classes

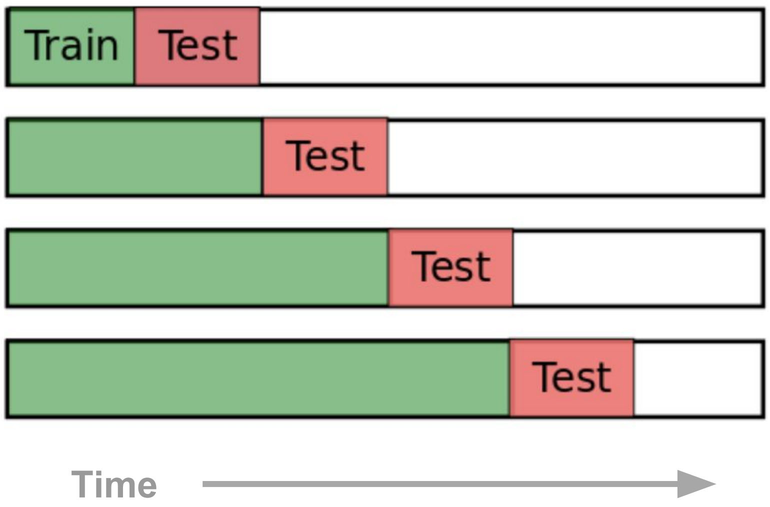
- New categories will be created / removed
- Proportions of classes can change by orders of magnitude (a problem for imbalanced data in general)
- The distributions on individual classes may also change

Bad assumption



Take a chronological test set

If you believe there is dataset shift, take your test set(s) based on *chronology*, you will get a more accurate picture



How do we fix this?

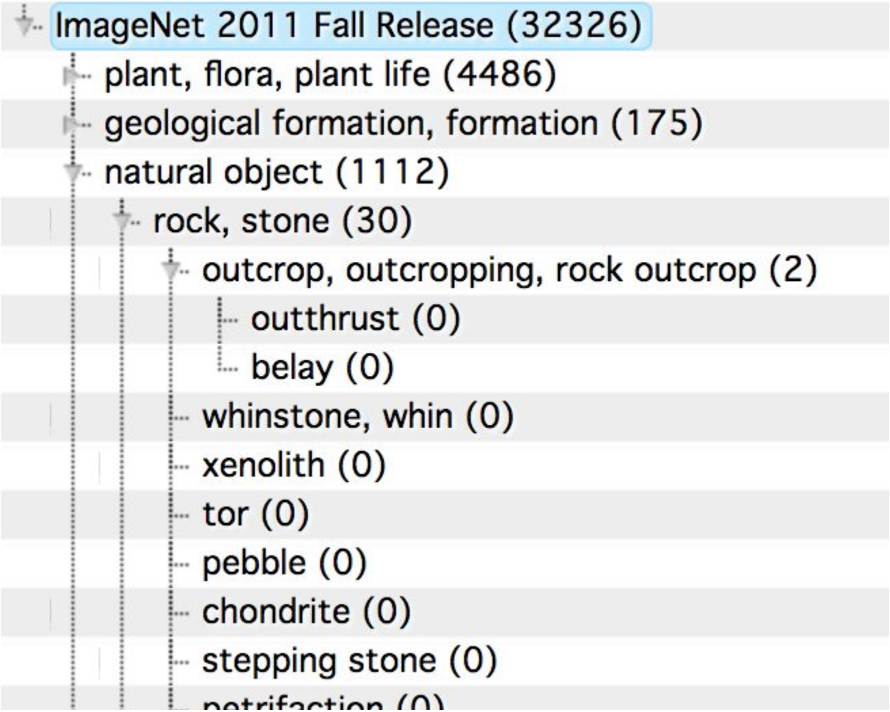
Ideal solution : keep labelling at least some of your data forever... At worst, this will give you a feel for how much your data is changing.

Other things you can do :

- try to predict how proportions are changing (dangerous)
- Train / calibrate models to imbalanced metrics

TO NOTEBOOK (?)

Quickfire round : hierarchies



Quickfire round : ensembling

Ensembling = combining the results of a set of different ML algorithms using a simple ML algorithm (e.g. logistic regression)

Harder when you have lots of classes because your ensemble algorithm will not be simple!

Ideas : simpler ensembling approaches, mixtures of experts [], using domain knowledge

Quickfire round : extreme classification

There is a lot of research around building classifiers that can deal with up to millions of classes (extreme classificatio / XML) [NIPS17].

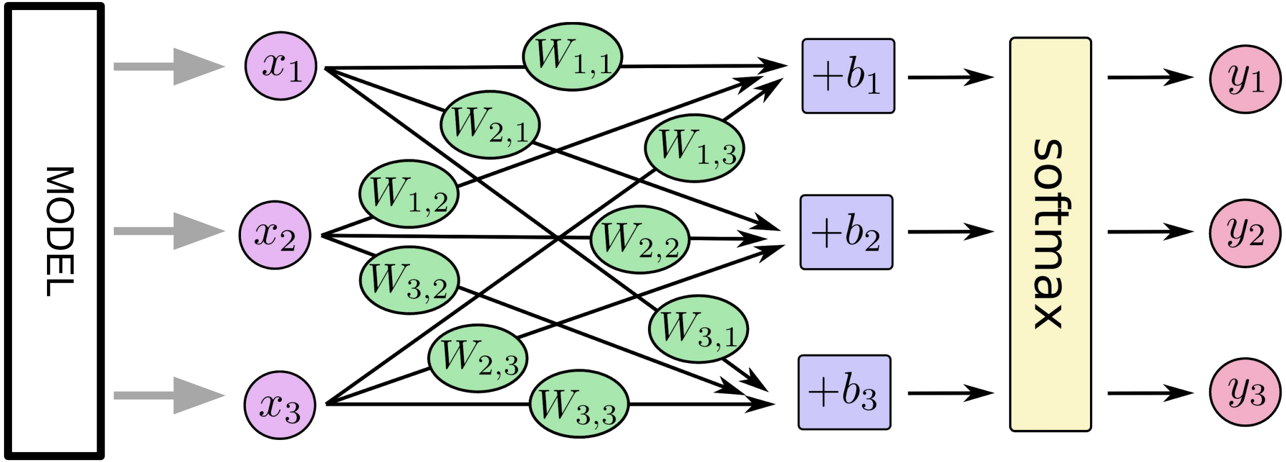
2 main categories of models

- Tree based approaches (e.g. FastXML [PV14])
- Embedding approaches (e.g. SLEEC [PJKVK15])

Word of warning : can be very susceptible to the datashift problem!

Quickfire round : Deep learning XML

Tends to be the same base models as normal deep learning (RNNs, CNNs, attention models etc.), the difference is in the final layer.

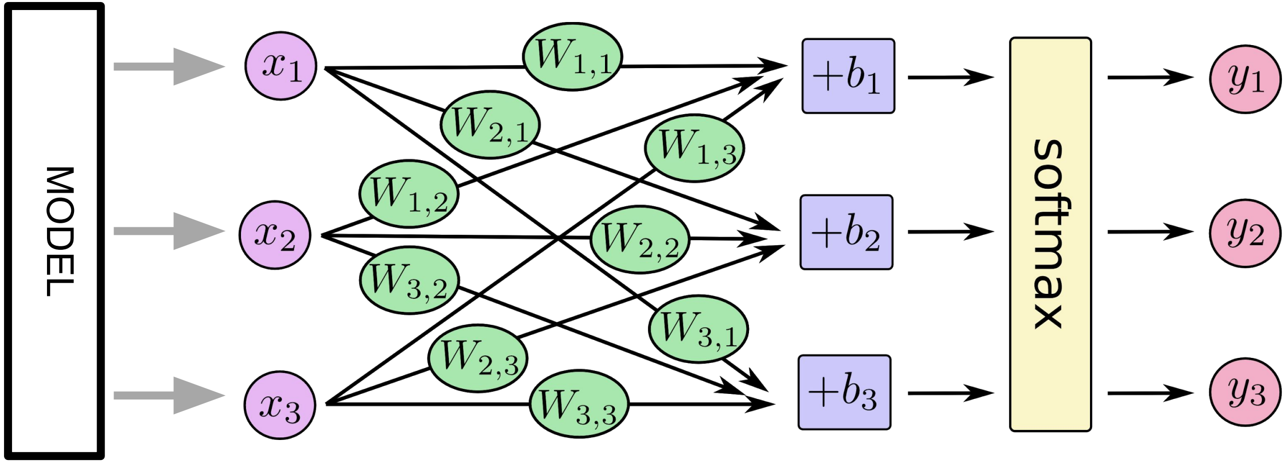


EvolutionAI

Quickfire round : Deep learning XML

Research tends to focus on either

- Approximating the softmax (e.g. hierarchical softmax, LSH)
- Embedding labels in some smaller space



Summary

- Avoid this problem if you can!
- Watch out for dataset shift
- Better feature selection always a good idea
- Look for ways to control complexity

Bibliography

[ILSVRC] ImageNet (wikipedia)
[JGBM16] Facebook FastText github
[JRT15] JRT paper
[KNH09] CIFAR homepage
[G18] Google text classification model selection advice
[HVD15] Distilling the knowledge in a neural network
[MH08] t-SNE paper
[NIPS 17] NIPS extreme classification track 2017
[PJKVK15] SLEEC
[PV14] FastXML
[SJ16] Rapbot blog post
[YGXWQ13] Imbalanced feature selection for imbalanced data