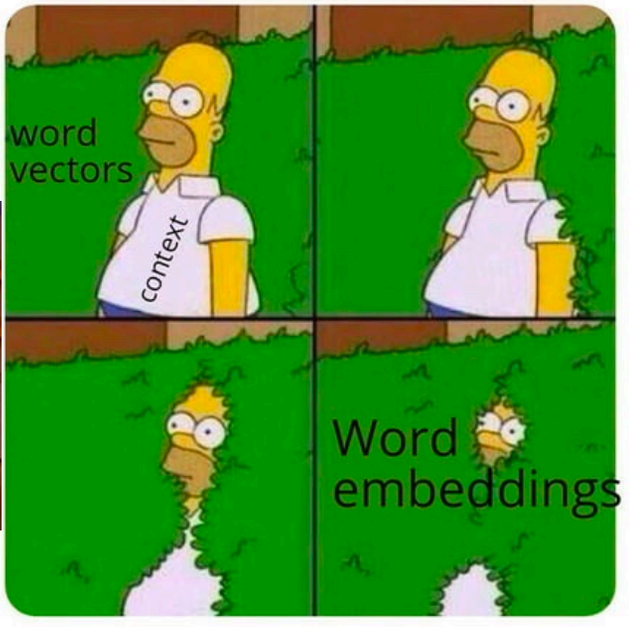
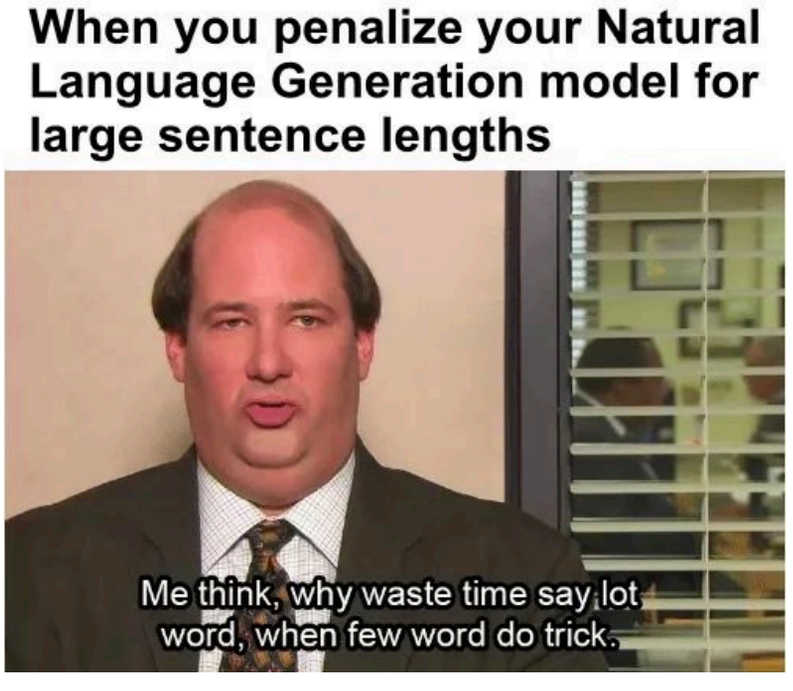
April 21, 2023.

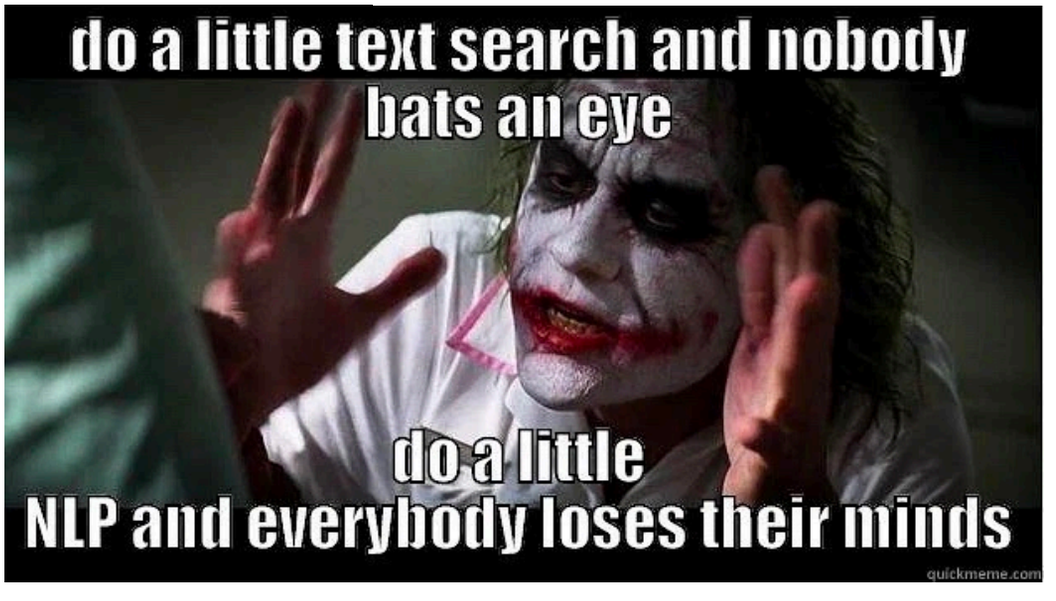
DSML : NLP module.

Text Representations.

Class starts @ 9:05







Announcements :

Hey Everyone,
On 22nd April at 03:00 PM IST, we have organised a session with one of yours peers where we will be discussing on the roadmap to crack Data Analyst Offers.
More details in the link to register.
Link to register - https://bit.ly/3UXzE2P
Regards,
Scaler Community

From learners: Could we have a similar event for Data Scientist roles?

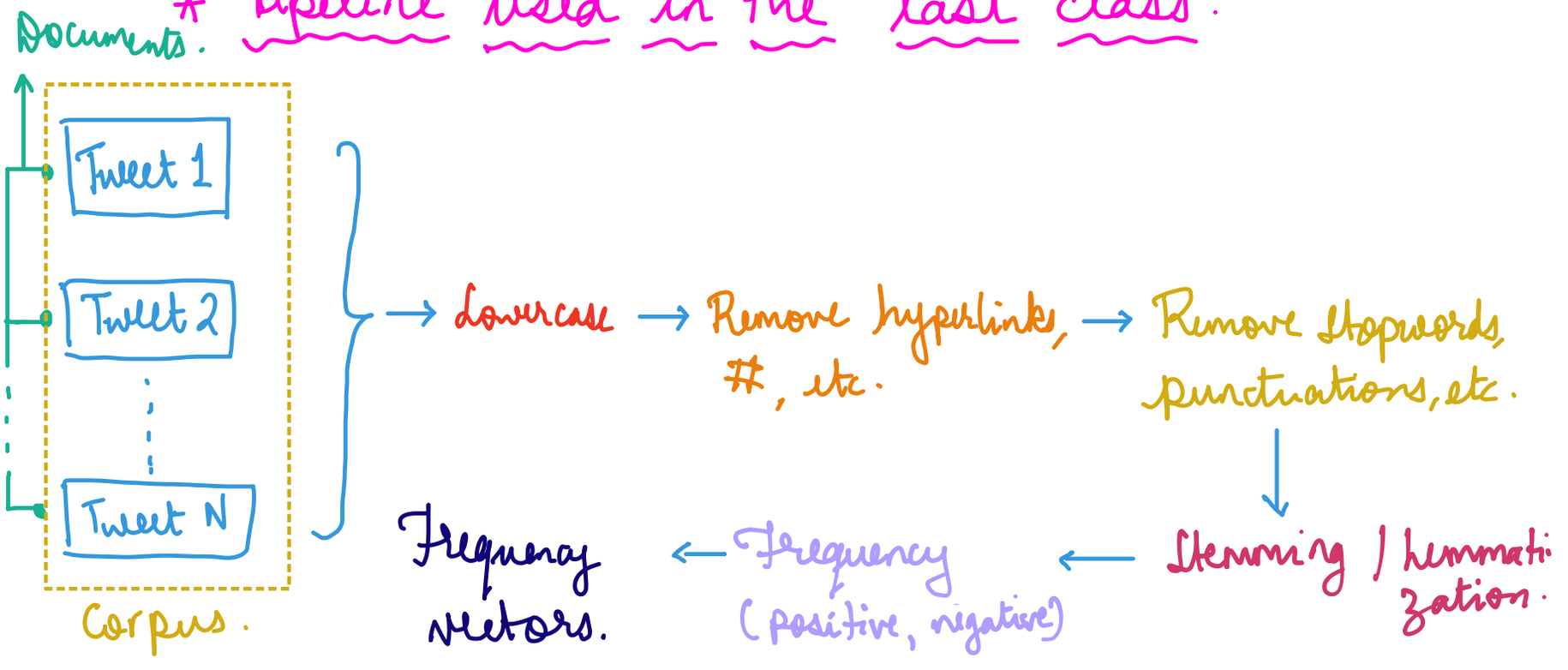
Recap:

* NLP → Natural language processing.

* Core Issue: Preprocessing.

"How to convert text to vectors?"

* Pipeline used in the last class:



Agenda: Text → Numerical vectors.

- BoW: Bag of words.

- TF-IDF: Term frequency, Inverse Document Frequency.

- Code: Finding similar blogs.

- T-SNE for data insights.

- Case : Finding similar blogs.

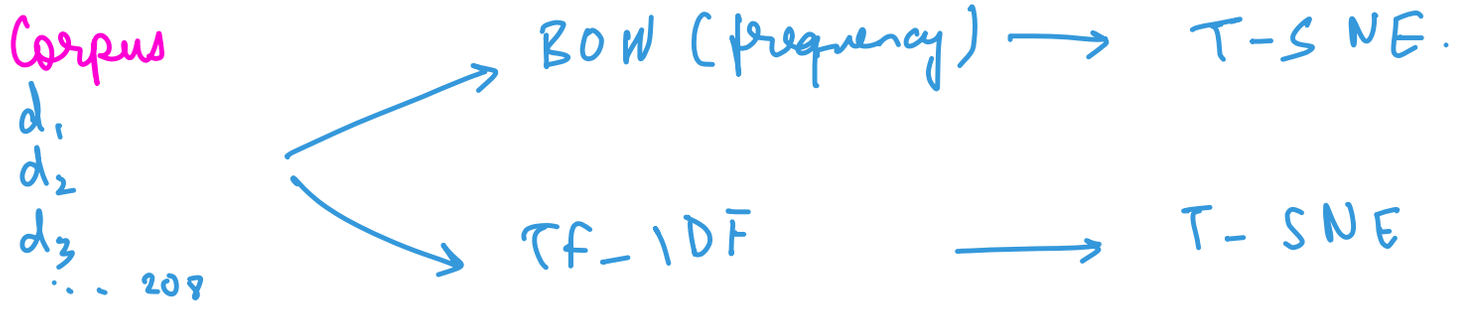
Finding similar Medium articles

`You are working as a Data Scientist at Medium`

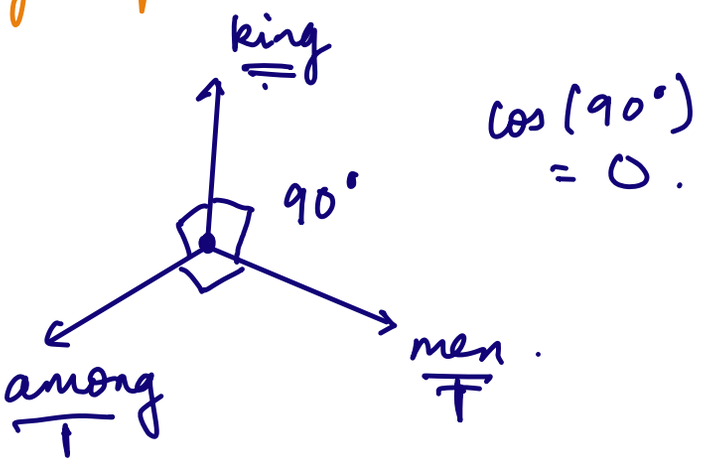
- Medium is an online publishing platform which hosts a hybrid collection of blog posts from both amateur and professional people and publications.
- In 2020, about 47,000 articles were published daily on the plaform and it had about 200M visitors every month.

Problem Statement:

- You want to give readers a better reading experience at Medium. To do that, you want to `recommend articles to the user on the basis of current article` that the user is reading.
- **More concretely, given a Medium article find a set of similar articles.**



The quick brown fox jumped over the lazy dog.



$\cos(90°) = 0$.

One-hot encoding.

[1 1 1 1]

The quick . . . . .

→ Sparsity.
→ All words have weightage!
→ Similar words are not close by.

He is a king among men.

[1, 0, 0] → [0, 0, 1]
[0, 1, 0] 0

Bag-of-words representations.

Corpus - doc1
doc2
⋮
docn.

0. pre-processing pipeline - lowercase, stop-word removal, stem/lem etc.
1. get all unique words from the corpus.
2. Put these words as the column names.
3. For each doc, find frequency & store in matrix.

Unresolved issues with BOW:

1. Weightage → frequency ✗
2. Cosine similarity.

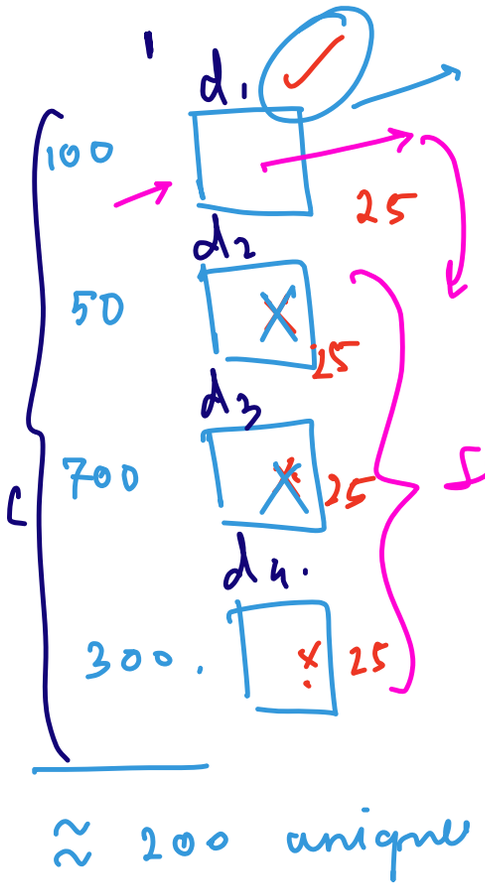
Document → Deep learning paper ✓

neural — 800

network — 1000

residual — 50

↑ Gives us some important info.



ResNet.

similar but more important.

similar.

Deep learning papers which apply VGG 16.

≈ 200 unique

$$IDF(t, d, C) = log(\frac{|C|}{|d \in C : t \in d|})$$

0.6

$$IDF(\text{"residual"}, d_1, C) = \log\left(\frac{4}{1}\right) = \log(4)$$

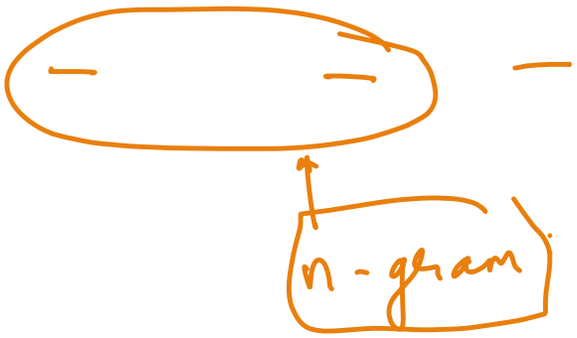
$$IDF(\text{"network"}, d_1, C) = \log\left(\frac{4}{4}\right) = \log(1) = 0.$$

"The new iPad is not good compared to previous releases."

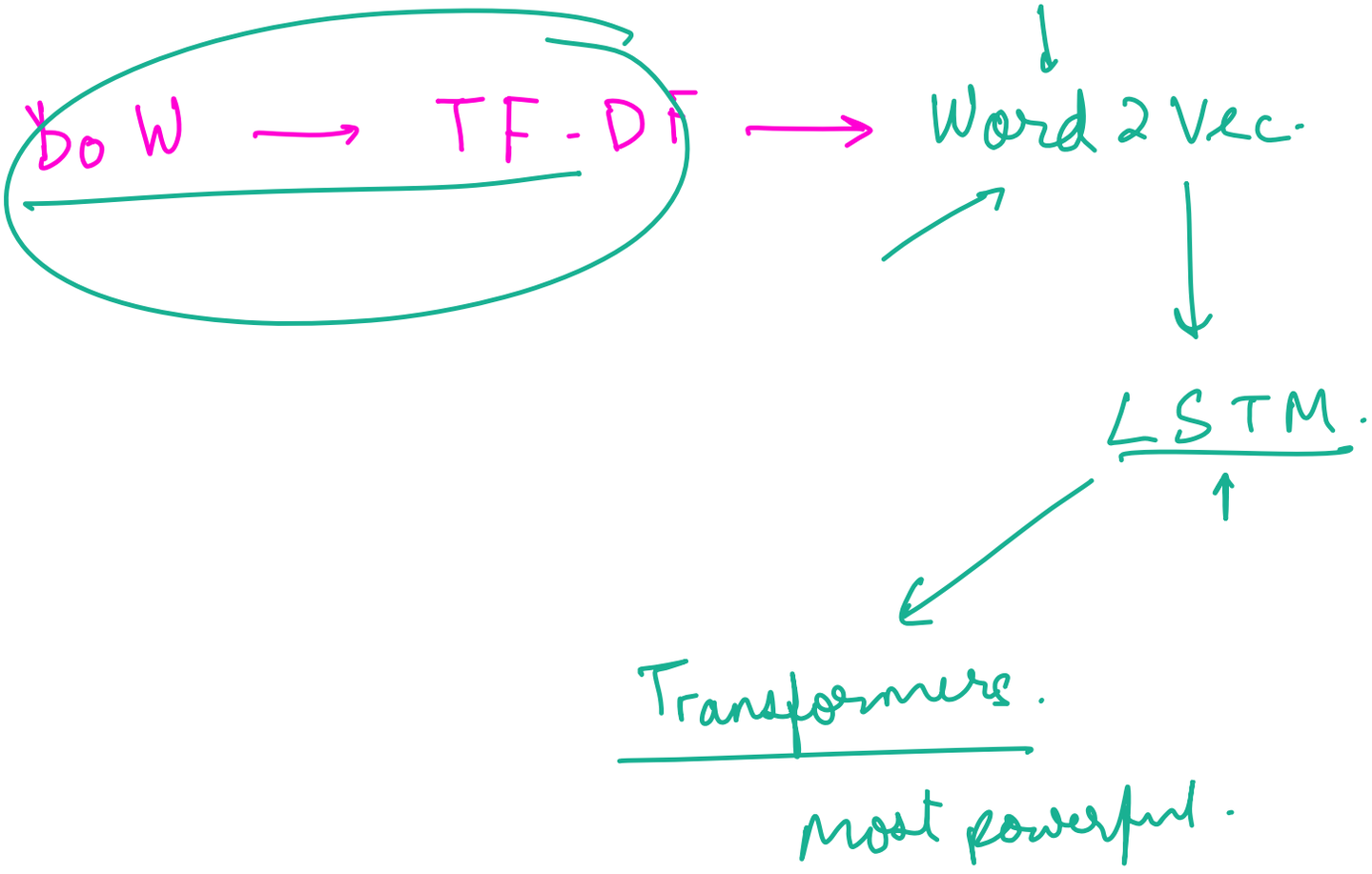
good x bad

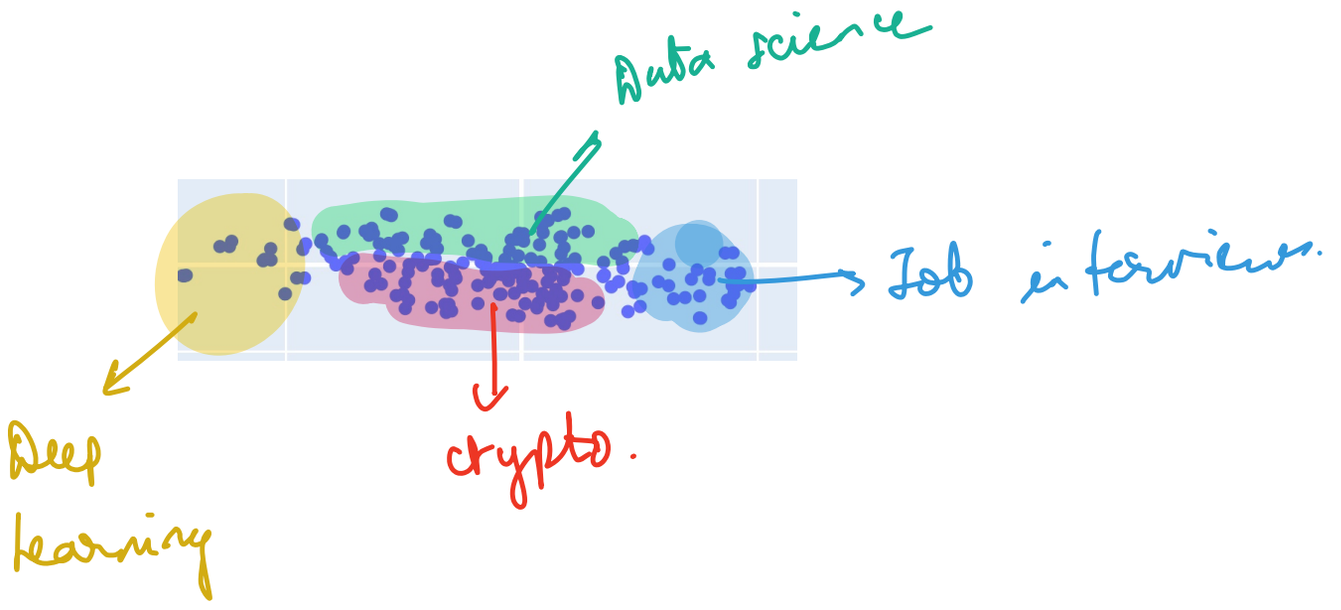
not good

n = 2



Instead of taking individual words, take pairs, triplet, quadruplet etc.


BoW
TF-IDF
Word2Vec.
LSTM.
Transformers.
most powerful.


Data science
Job interviews.
Deep learning
crypto.

Concluding Remarks:

* How to convert text → Vectors.

* We studied 2 methods - BoW, TF-IDF.

* BoW → TF-IDF

  BoW: only document level features

  TF-IDF: Allows us to compare documents to whole corpus.

* Shortcomings:
  1. Context linkage missing.
  2. Size of the vector depends on unique words.

IDF → |C| → # of docs in corpus.

2. Max IDF,

$$IDF(t, C) = log(\frac{max_{t' \in d}(|d \in C : t' \in d|)}{|d \in C : t \in d|})$$

3

$\max_{t' \in d} \left(|d \in C : t' \in d| \right)$

How many documents does the most common word appear in

over all docs in C, find if t' belongs to D.

"residual"