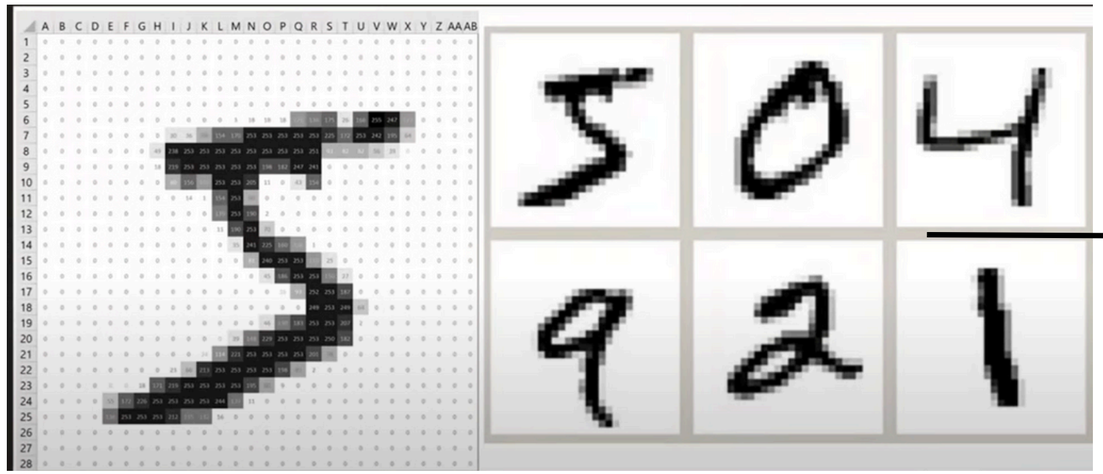
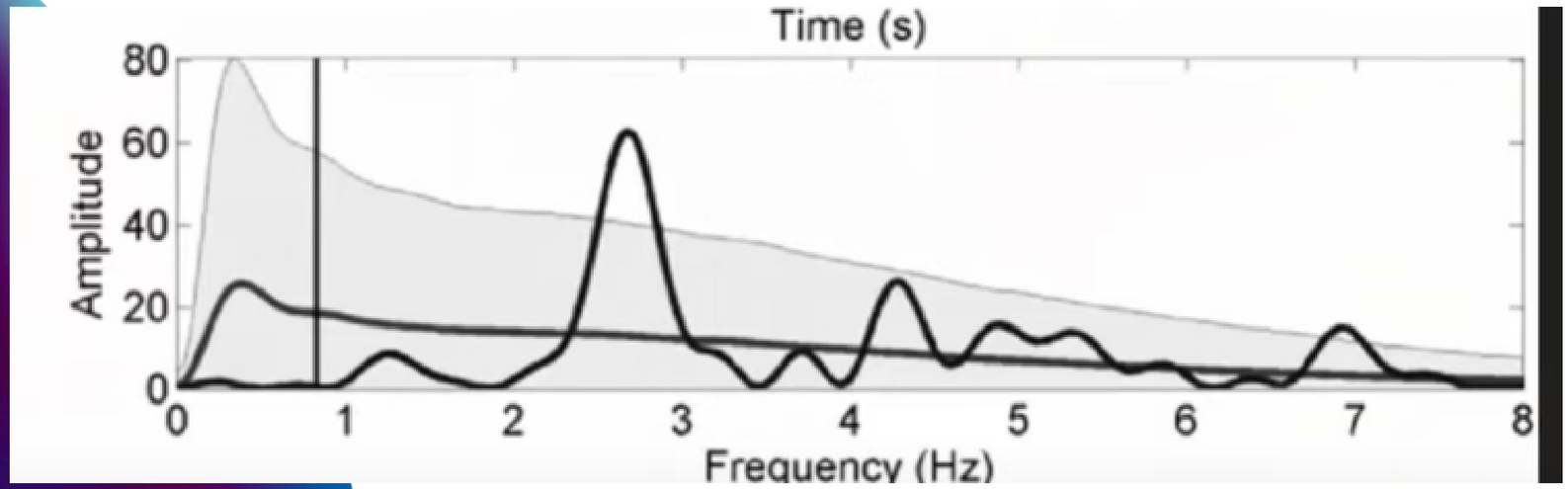
TF-IDF

Image



$$\begin{pmatrix} 255 \\ 231 \\ 42 \\ 22 \\ 123 \\ 94 \\ \vdots \\ \vdots \\ 92 \\ 142 \end{pmatrix}$$

Audio

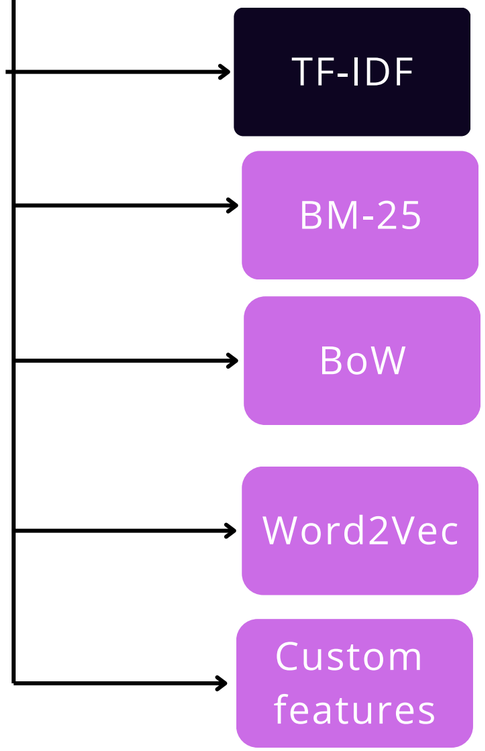


$$\begin{bmatrix} 10 \\ 20 \\ 10 \\ 14 \\ 0 \\ 5 \\ 80 \\ 90 \end{bmatrix}$$

How can you vectorize a text lets say "Hello, How are you?"

How do you extract meaning from "just plain" text?

Basic Methods of Text Vectorization



Time Line of text vectorization

BoW — Zellig Harris's 1954

TF-IDF — 1972 by Salton and McGill

BM-25 — Robertson and Walker in 1994

Transformers — 2017 by Google AI researchers

Terminology

04

Term Frequency

frequency of a particular term relative to the document - raw count, adjusted for length, logarithmic, boolean, etc

Corpus

A collection of documents

Inverse document frequency

the measure of how common a term is across documents

We will use these two statements as an example

A: “The car is driven on the road.“

B: “The truck is driven on the highway”

A and B are individually called a document.

Collection of all words of the A and B (including repetition) is corpus.

Term Frequency

Definition:

How often a term appears in a document compared to the total number of terms.

Formulae:

$$\mathrm{TF}(t, d) = \frac{\text{Count of term } t \text{ in document } d}{\text{Total terms in document } d}$$

A: **“The car is driven on the road.“**

B: **“The truck is driven on the highway”**

Example:

Word	TF	
	A	B
The	1/7	1/7
Car	1/7	0
Truck	0	1/7
Is	1/7	1/7
Driven	1/7	1/7
On	1/7	1/7
The	1/7	1/7
Road	1/7	0
Highway	0	1/7

Variants of term frequency (tf) weight

weighting scheme	tf weight
binary	$0, 1$
raw count	$f_{t,d}$
term frequency	$f_{t,d} \Big/ \sum_{t' \in d} f_{t',d}$
log normalization	$\log(1 + f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K) \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

Inverse Document Frequency

Definition:

How unique or rare a term is across the entire corpus.

Formulae:

$$\text{IDF}(t) = \log \frac{\text{Total documents}}{\text{Number of documents containing term } t}$$

Example:

Word	TF		IDF
	A	B	
The	1/7	1/7	log(2/2) = 0
Car	1/7	0	log(2/1) = 0.3
Truck	0	1/7	log(2/1) = 0.3
Is	1/7	1/7	log(2/2) = 0
Driven	1/7	1/7	log(2/2) = 0
On	1/7	1/7	log(2/2) = 0
The	1/7	1/7	log(2/2) = 0
Road	1/7	0	log(2/1) = 0.3
Highway	0	1/7	log(2/1) = 0.3

Variants of inverse document frequency (idf) weight

weighting scheme	idf weight ($n_t = \left\vert\{d \in D : t \in d\}\right\vert$)
unary	1
inverse document frequency	$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$
inverse document frequency smooth	$\log\left(\frac{N}{1+n_t}\right)+1$
inverse document frequency max	$\log\left(\frac{\max_{\{t' \in d\}} n_{t'}}{1+n_t}\right)$
probabilistic inverse document frequency	$\log \frac{N-n_t}{n_t}$

Combined TF-IDF

Combined TF-IDF

$$TF\text{-}IDF = \left(\frac{\text{Term Count in Doc}}{\text{Total Terms}}\right) \times \log\left(\frac{\text{Total Docs}}{\text{Docs with Term}}\right)$$

Word	TF		IDF	TF*IDF	
	A	B		A	B
The	1/7	1/7	log(2/2) = 0	0	0
Car	1/7	0	log(2/1) = 0.3	0.043	0
Truck	0	1/7	log(2/1) = 0.3	0	0.043
Is	1/7	1/7	log(2/2) = 0	0	0
Driven	1/7	1/7	log(2/2) = 0	0	0
On	1/7	1/7	log(2/2) = 0	0	0
The	1/7	1/7	log(2/2) = 0	0	0
Road	1/7	0	log(2/1) = 0.3	0.043	0
Highway	0	1/7	log(2/1) = 0.3	0	0.043

Can you Combine any TF variant with any IDF variant?

Yes, because both terms are independent and give scalar weights so it is also mathematically valid.

Binary TF + IDF

$$\text{TF} - \text{IDF} = (0 \text{ or } 1) \cdot \log\left(\frac{N}{n_t}\right)$$

- Useful in document classification where only the presence or absence of terms matters.
- Example: Identifying whether a document contains specific keywords.

Raw Count TF + Smooth IDF

$$\text{TF} - \text{IDF} = f_{t,d} \cdot \left(\log\left(\frac{N}{1+n_t}\right) + 1\right)$$

- Suitable for text analysis when raw term frequencies are important but need smoothing to avoid division by zero.
- Example: Ranking documents by term relevance in a small corpus.

Sci-kit Learn Version of TF-IDF

$$\mathrm{TF}(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$$

$$\mathrm{IDF}(t) = \log\left(\frac{1+n}{1+\mathrm{df}(t)}\right) + 1$$

$$\text{TF-IDF}(t,d) = \mathrm{TF}(t,d) \cdot \mathrm{IDF}(t)$$

- Ensures no division by zero (if a term is missing in all documents).
- Helps avoid extreme IDF values when df(t) is very low.

Limitations

01

overvalues rare words, undervalues important but frequent words

02

no semantic relevance eg - apple, Apple Inc

03

cannot recognize context spread over many words

Solutions

Pre - processing

01

N - grams

02

Lemmatization

03

Stop words

04

Synonym handling

N-grams

An n-gram is a continuous sequence of ‘n’ words in a text

Ex - Will you go to prom with me?

Unigram - [’Will’ , ‘you’, ‘go’, ‘to’, ‘prom’, ...]

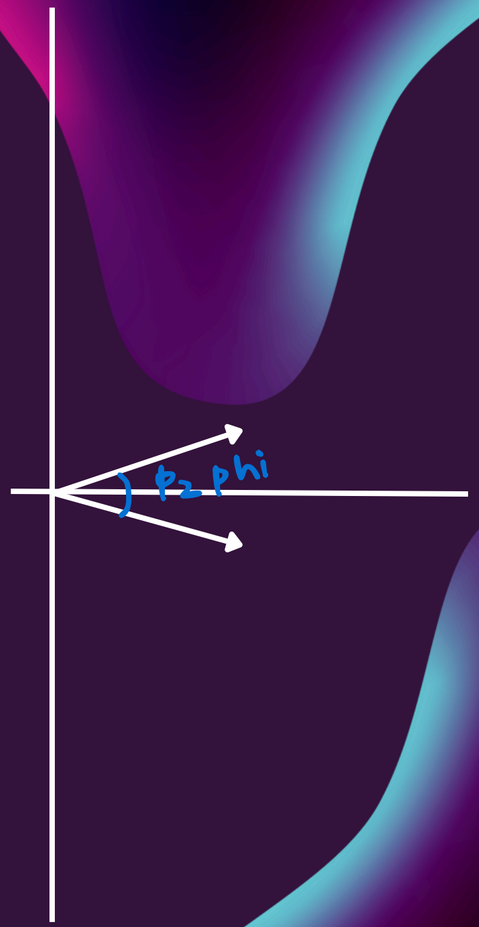
Bigram - [’Will you’, ‘you go’, ‘go to’, ‘to prom’,...]

Trigram - [’Will you go’, ‘you go to’, ‘go to prom’...]

A: This is a good book

B: This is not a good book

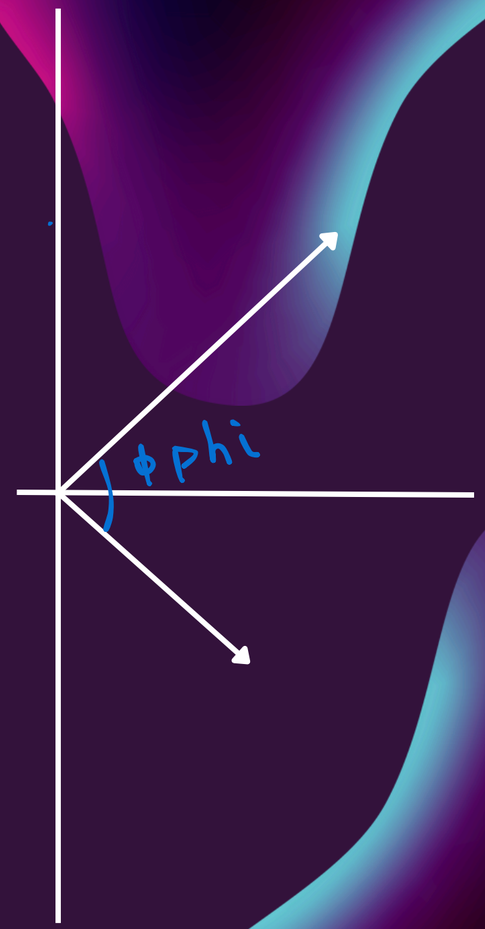
unigram	This	is	a	good	book	not
count	1	1	1	1	1	0
unigram	This	is	a	good	book	not
count	1	1	1	1	1	1



A: This is a good book

B: This is not a good book

bigram	This is	is a	a good	good book	is not	not a
count	1	1	1	1	0	0
bigram	This is	is a	a good	good book	is not	not a
count	1	1	1	1	1	1



This phi is larger than other phi

Lemmatization

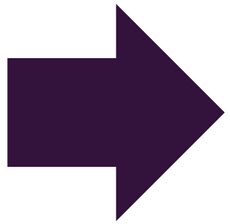
Lemmatization is the process of reducing a word to its base or dictionary form (lemma) while considering its meaning and context.

Word Form	Part of Speech	Lemma
Loves	Verb (third-person singular)	**Love**
Loving	Verb (present participle)	**Love**
Loved	Verb (past tense)	**Love**
Loving	Adjective	**Loving** (remains unchanged)

Stop words

Stop words are common words that are usually removed from text during natural language processing (NLP) because they don’t carry significant meaning for analysis

The quick brown fox jumps over the lazy dog while searching for food in the forest. It is important to note that this clever animal often hunts at dawn, using its sharp senses to locate small prey hiding under leaves or between rocks near shallow streams.



quick brown fox jumps lazy dog searching food forest .
important note clever animal often hunts dawn , using sharp senses locate small prey hiding leaves rocks near shallow streams .

Before Pre-processing

```
from sklearn.ensemble import RandomForestClassifier

#1. create a pipeline object
clf = Pipeline([
     ('vectorizer_tfidf',TfidfVectorizer()),        #using the ng
     ('Random Forest', RandomForestClassifier())
])

#2. fit with X_train and y_train
clf.fit(X_train, y_train)


#3. get the predictions for X_test and store it in y_pred
y_pred = clf.predict(X_test)


#4. print the classfication report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.96      0.96      0.96      1200
           1       0.98      0.98      0.98      1200
           2       0.98      0.97      0.97      1200
           3       0.98      0.99      0.98      1200

    accuracy                           0.97      4800
   macro avg       0.97      0.97      0.97      4800
weighted avg       0.97      0.97      0.97      4800
```

After Pre-processing

	precision	recall	f1-score	support
0	0.96	0.96	0.96	1200
1	0.98	0.98	0.98	1200
2	0.98	0.98	0.98	1200
3	0.98	0.99	0.99	1200
accuracy			0.98	4800
macro avg	0.98	0.98	0.98	4800
weighted avg	0.98	0.98	0.98	4800

When to Use TF-IDF

- Short-to-Medium Texts
- Social media posts
- Product reviews
- Chat conversations

THANKS

Credits

Makers - Aishwarya and Soham
Powered by - AI Club
Co-powered by - CFI
In association with IIT Madras
Special thanks to Anish and Nishitha for conducting this project

We apologize if there are any mistakes!!!