NLP

Vectorización de documentos

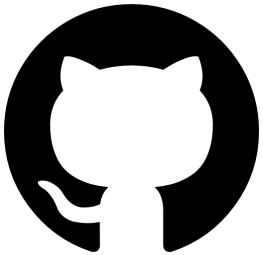
Msc. Rodrigo Cardenas Szigety
rodrigo.cardenas.sz@gmail.com

Programa de la materia

Clase 1: Introducción a NLP, Vectorización de documentos.
Clase 2: Preprocesamiento de texto, librerías de NLP y Rule-Based Bots.
Clase 3: Word Embeddings, CBOW y SkipGRAM, representación de oraciones.
Clase 4: Redes recurrentes (RNN), problemas de secuencia y estimación de próxima palabra.
Clase 5: Redes LSTM, análisis de sentimientos.
Clase 6: Modelos Seq2Seq, traductores y bots conversacionales.
Clase 7: Celdas con Attention. Transformers, BERT & ELMo, fine tuning.
Clase 8: Cierre del curso, NLP hoy y futuro, deploy.

*Unidades con desafíos a presentar al finalizar el curso.
*Último desafío y cierre del contenido práctico del curso.

Link Github de la materia



https://github.com/FIUBA-Posgrado-Inteligencia-Artificial/procesamiento_lenguaje_natural

En el Github van a encontrar...

LINK

Los ejemplos de clase y de tarea están propuestos tanto en *Tensorflow* como en *Pytorch*

Pueden usar el framework que más cómodo les resulte

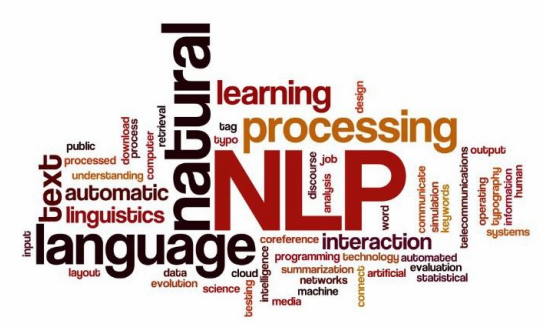
TensorFlow

- Creado por Google
- Utilizado principalmente en la industria y en el despliegue.
- Los bloques del framework son bastante cerrados.
- Posee muchas librerías y tools que de ayudan.
- Muchas tools para despliegue y debugging

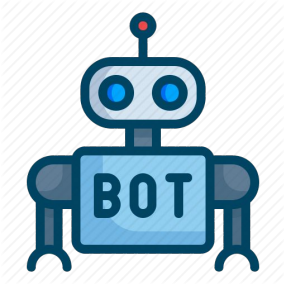
PYTORCH

- Creado por Facebook
- Utilizado principalmente en el campo académico e investigación.
- Los bloques del framework son totalmente abiertos.
- Posee pocas librerías o tools, hay que desarrollar mucho uno mismo.
- Los nuevos modelos de NLP salen antes en Pytorch que en Tensorflow

Desafíos semanales

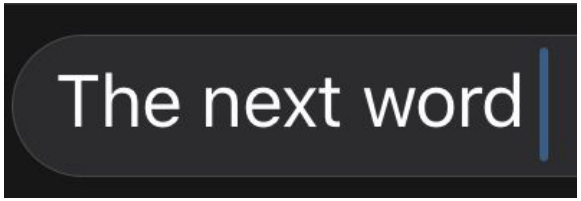


Vectorización de texto



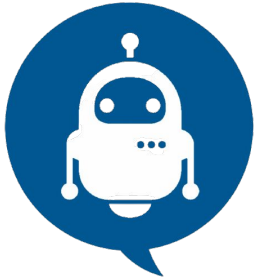
Bot "simple"

Word Embedding



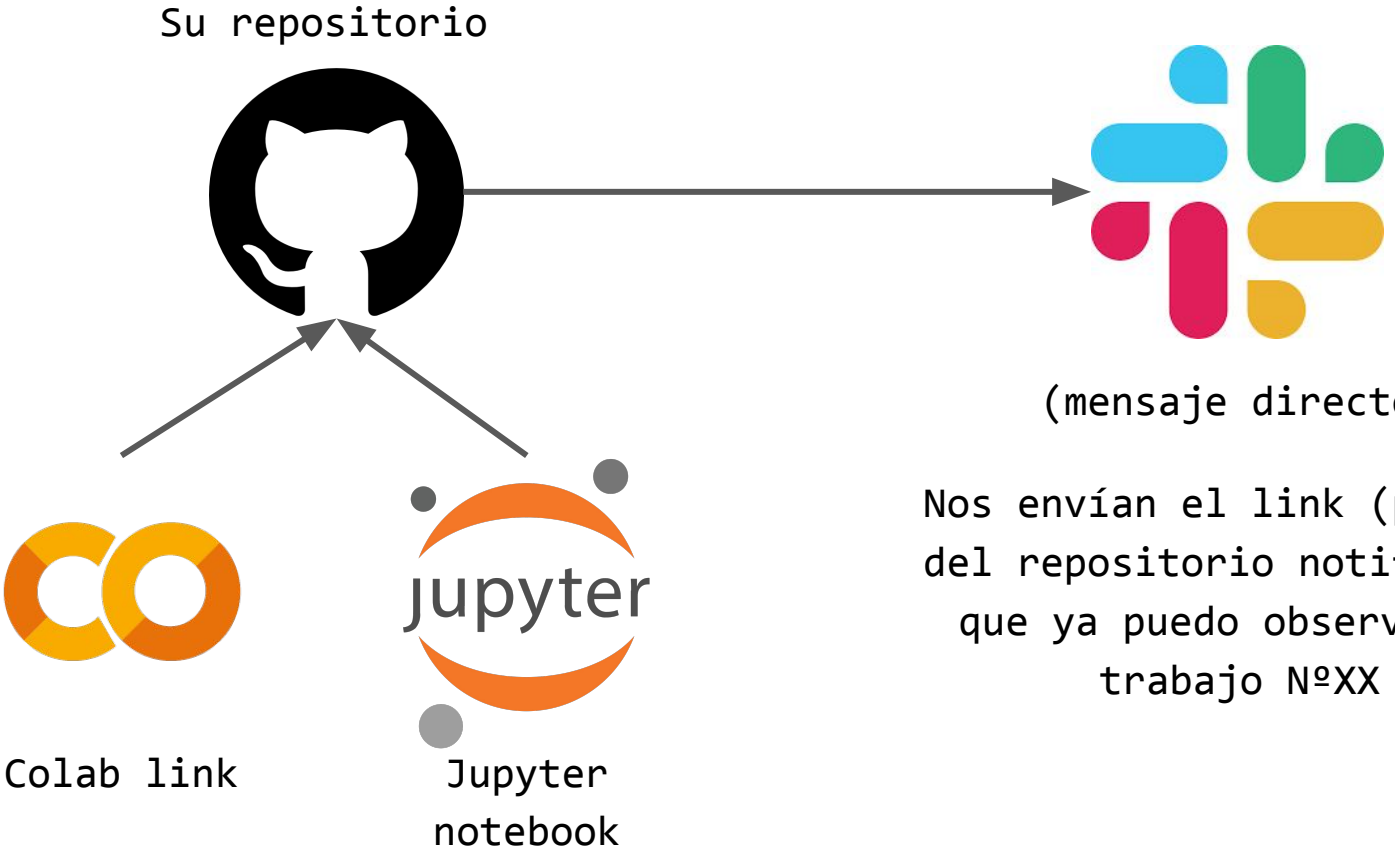
Predicción de
próxima palabra

Análisis de
sentimientos



Bot
conversacional

¿Cómo me acercaran sus soluciones?



¿Cómo se evaluarán los desafíos?

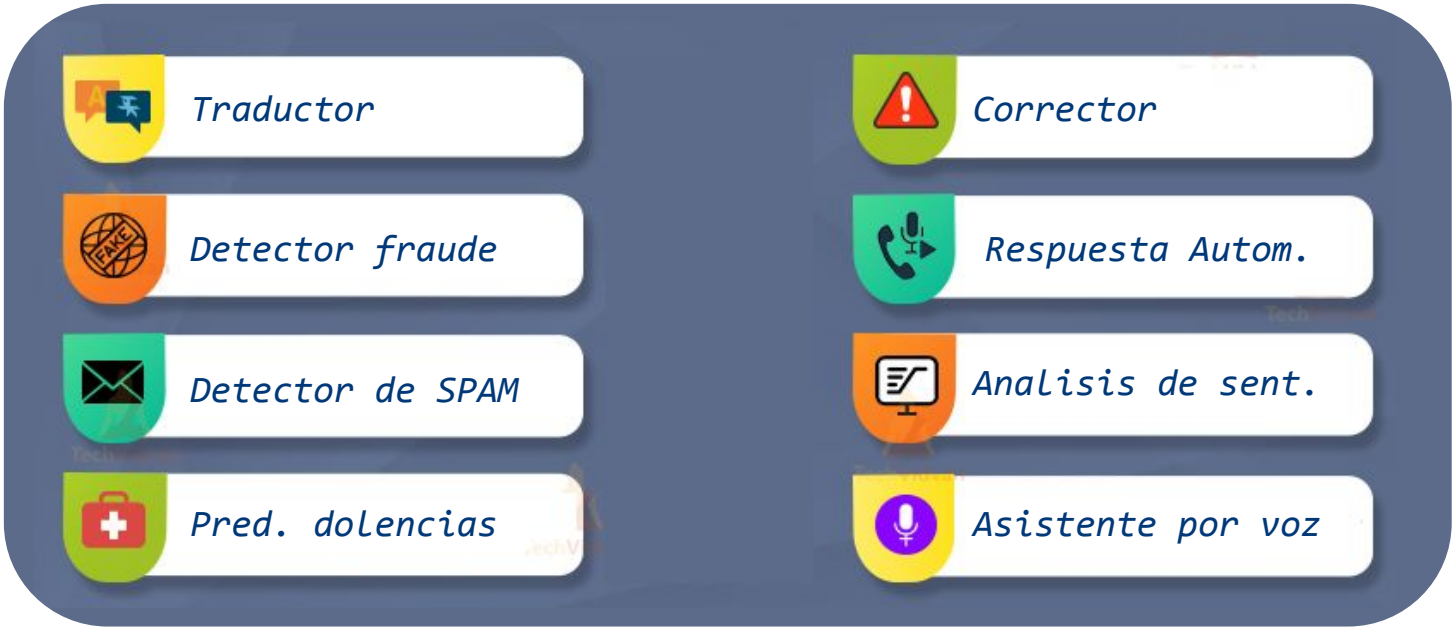
	Clases							Recu
	1~2	2~3	3~4	4~5	5~6	6~7	7~8	8	
Desafío 1	9-10	9-10	8-9	8-9	7-8	7-8	6-7	6-7	4-6
Desafío 2		9-10	9-10	8-9	8-9	7-8	7-8	6-7	4-6
Desafío 3			9-10	9-10	8-9	7-8	7-8	6-7	4-6
Desafío 4				9-10	9-10	8-9	7-8	6-7	4-6
Desafío 5					9-10	9-10	8-9	7-8	4-6
Desafío 6						9-10	8-9	7-8	4-6
Desafío 7							9-10	8-9	4-6

*La instancia de recuperación comienza 1 semana después de haberse dictado la última clase. La instancia de recuperación tiene una duración de una semana límite para terminar de entregar los desafíos

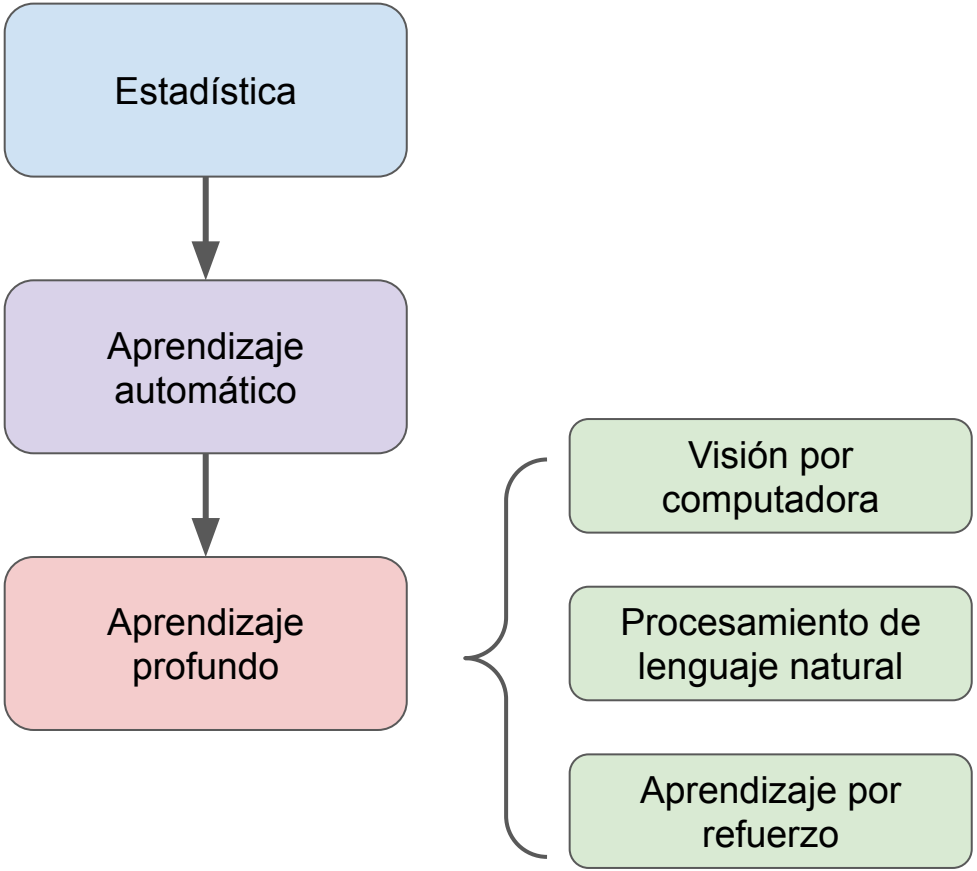
¿Qué es NLP?

El procesamiento de lenguaje natural (PLN o NLP) es un campo de la

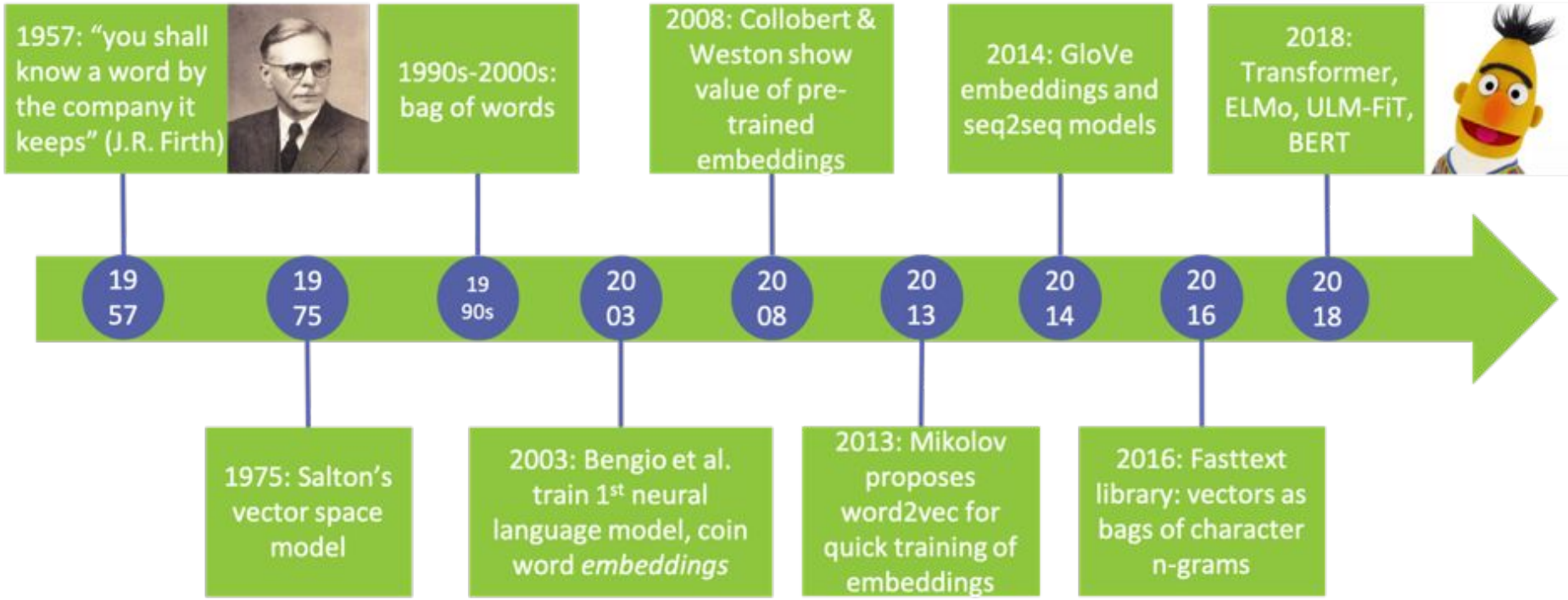
Inteligencia artificial + Lingüística



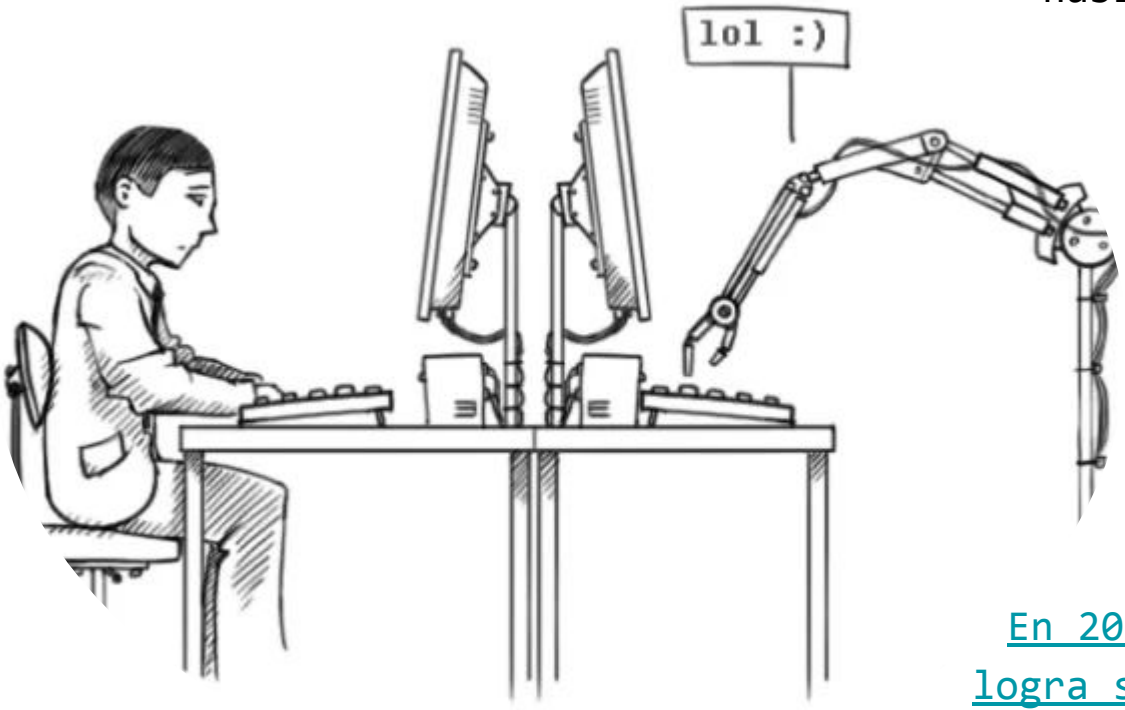
Campos de aplicación del data science



Timeline


1957: "you shall know a word by the company it keeps" (J.R. Firth)
1990s-2000s: bag of words
2008: Collobert & Weston show value of pre-trained embeddings
2014: GloVe embeddings and seq2seq models
2018: Transformer, ELMo, ULM-FiT, BERT
19 57
19 75
19 90s
20 03
20 08
20 13
20 14
20 16
20 18
1975: Salton's vector space model
2003: Bengio et al. train 1st neural language model, coin word *embeddings*
2013: Mikolov proposes word2vec for quick training of embeddings
2016: Fasttext library: vectors as bags of character n-grams

Test de turing "1950"

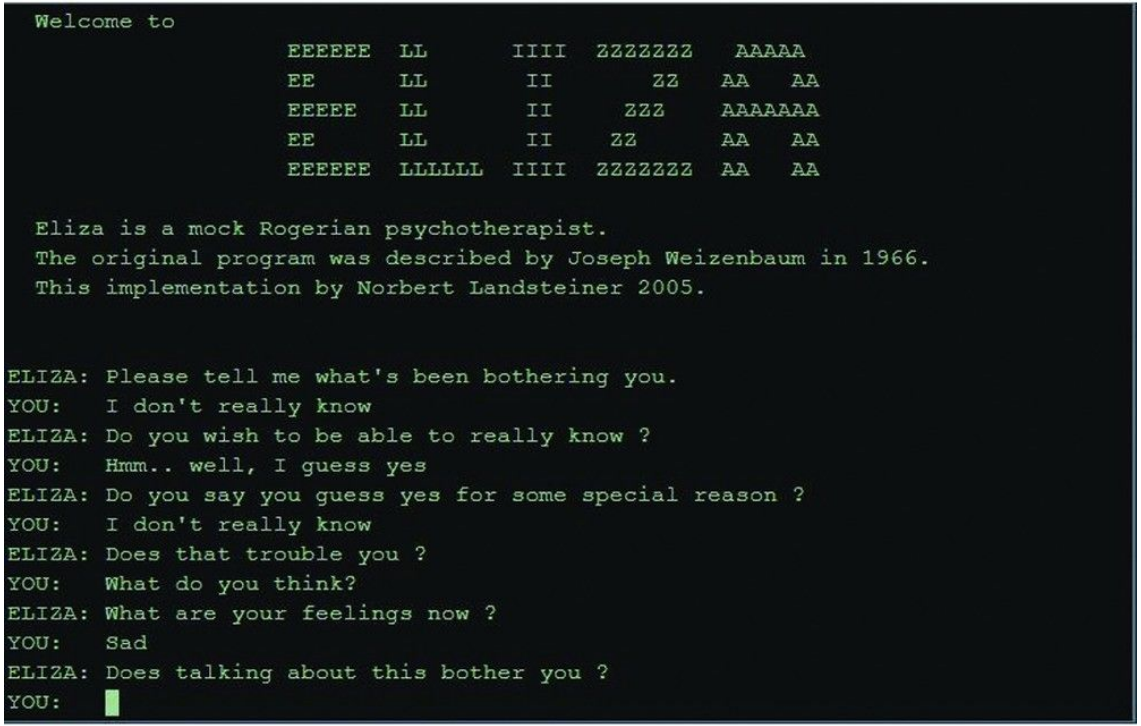


¿Puede una máquina hablar como si fuera un humano?

Blade runner
"Test emocional"

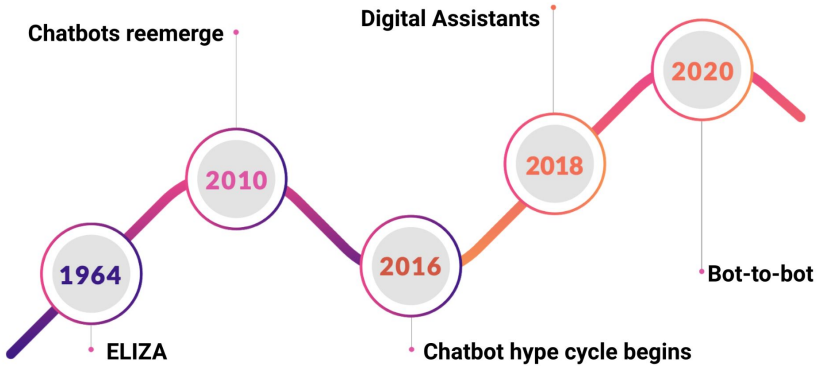
En 2014 una computadora logra superar parcialmente este ensayo

Eliza (1964) - Primer bot NLP



Se basaba en encontrar las palabras claves del texto y utilizarlas para formular una pregunta

Los chatbots y asistentes virtuales...



IBM Watson
2006

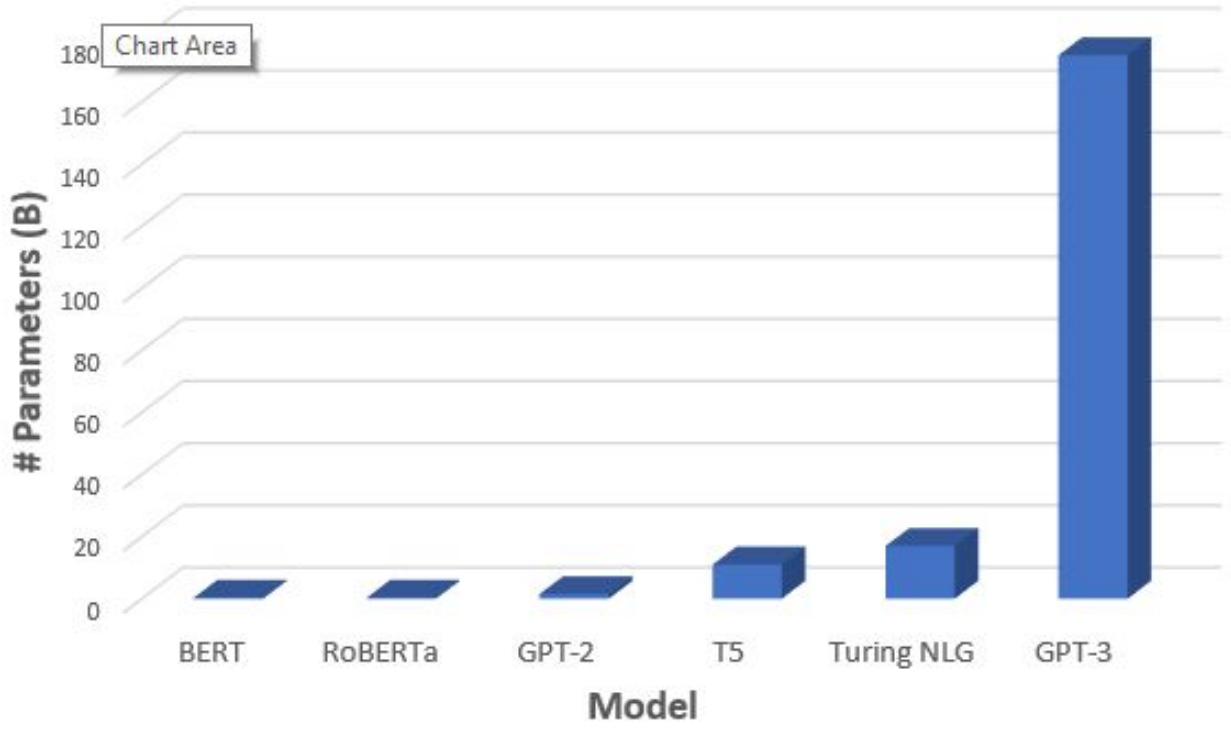
Apple Siri
2010

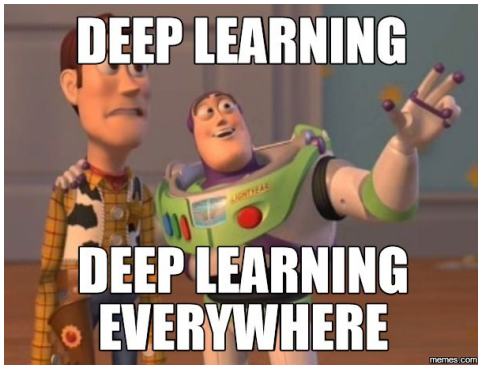
Google Now
2012

Amazon
alexa
2015

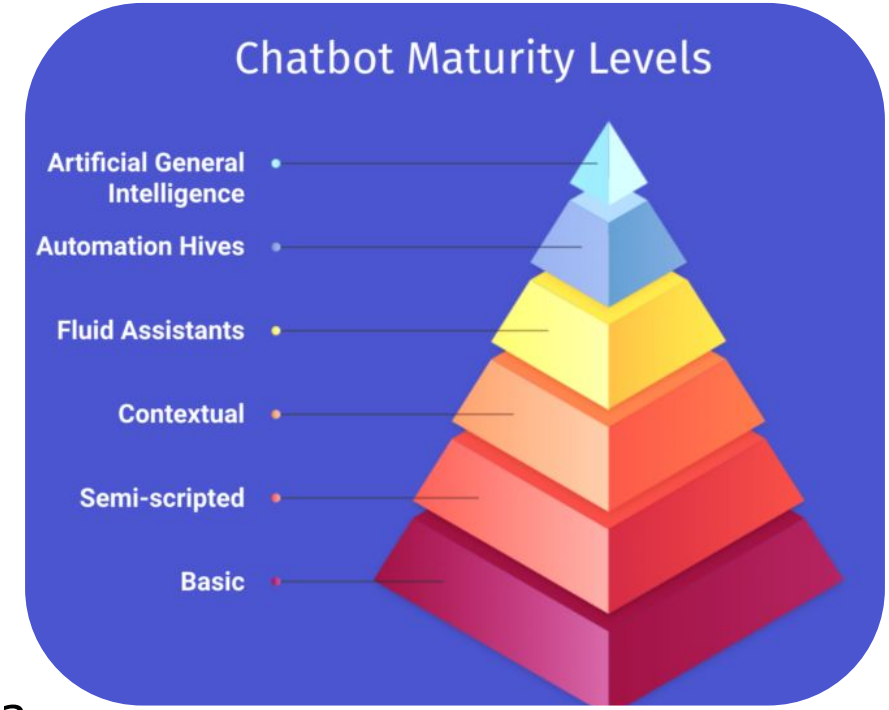
Microsoft
Cortana
2015

Los modelos que transformaron NLP





Elegir la herramienta que más se ajusta a sus problemas

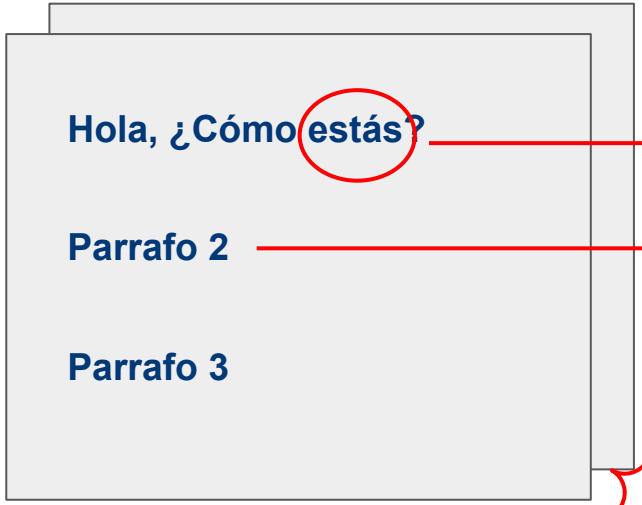


inteligencia

costo

Vectorización de texto

LINK GLOSARIO



Término t: palabra/símbolo "t" del documento

Document: su largo es variable, normalmente una sentencia/oración/párrafo.

Corpus: conjunto de documentos, forman todo el vocabulario.

No podemos ingresar texto a una red

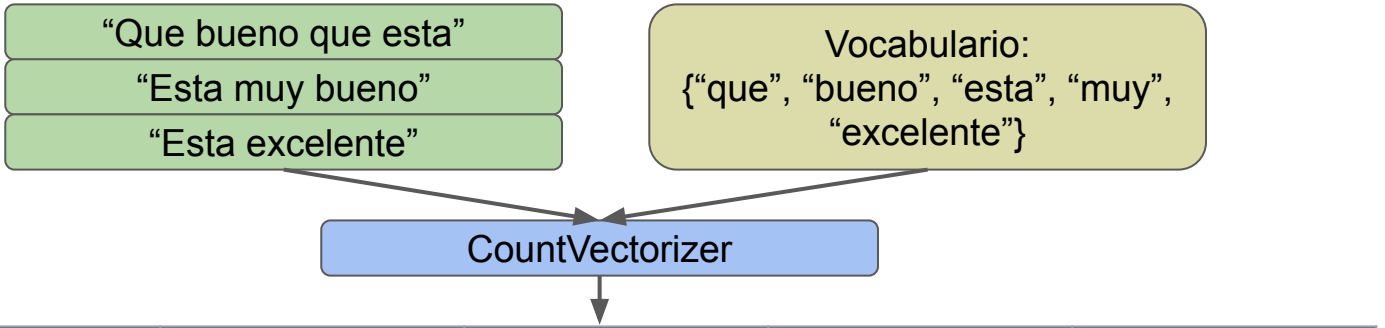
¿Cómo transformamos palabras a números?

vectorización

word2vect

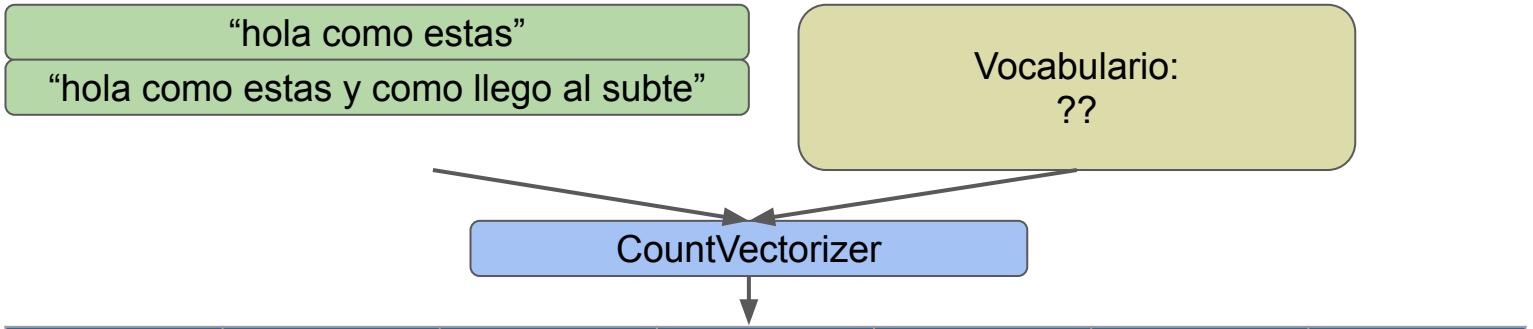
Vectores de frecuencia

"Por cada documento en el corpus se calcula un vector que representa cuántas veces cada palabra del vocabulario aparece en ese documento"



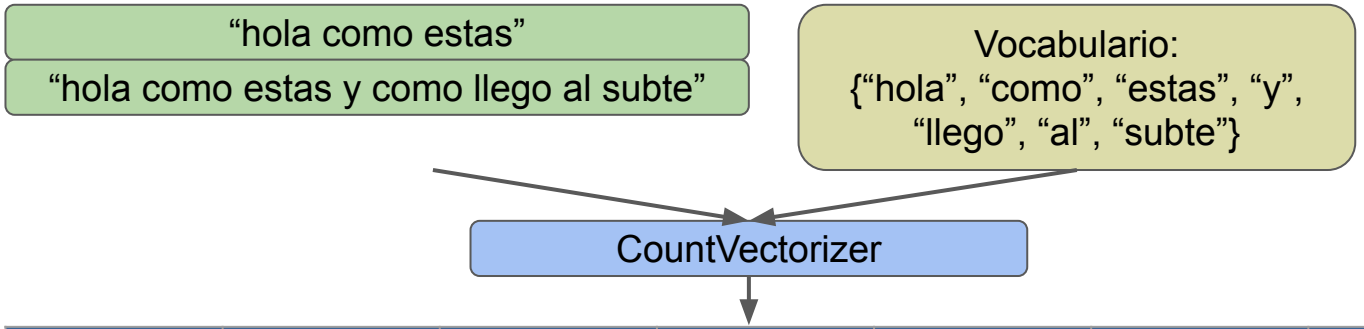
que	bueno	esta	muy	excelente
2	1	1	0	0
0	1	1	1	0
0	0	1	0	1

Vectores de frecuencia (ejemplo)



hola	como	estas	y	llego	al	subte

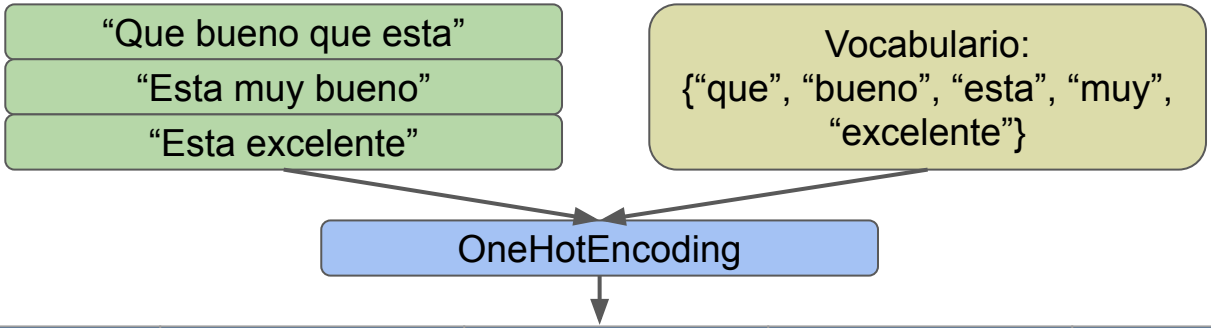
Vectores de frecuencia (ejemplo resuelto)



hola	como	estas	y	llego	al	subte
1	1	1	0	0	0	0
1	2	1	1	1	1	1

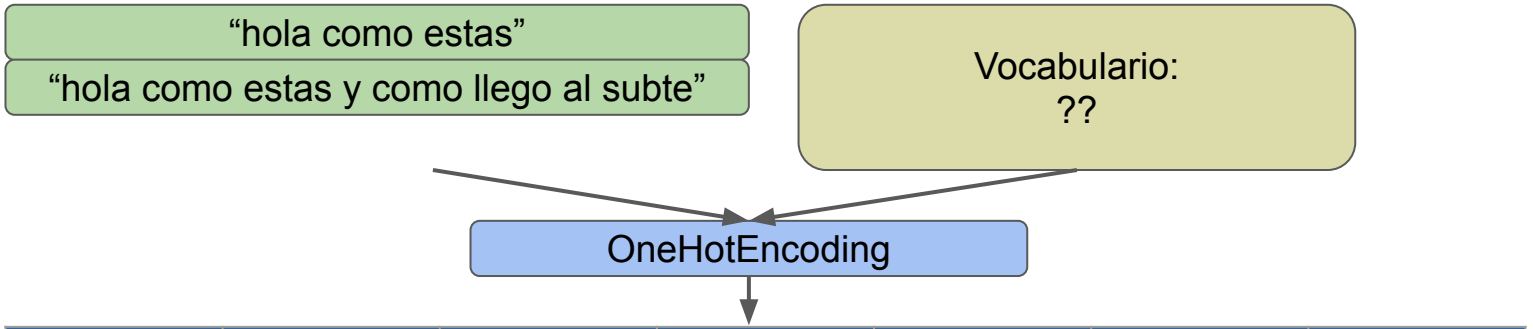
One-hot encoding

"Por cada documento en el corpus se calcula un vector que representa si cada palabra del vocabulario aparece o no en ese documento"



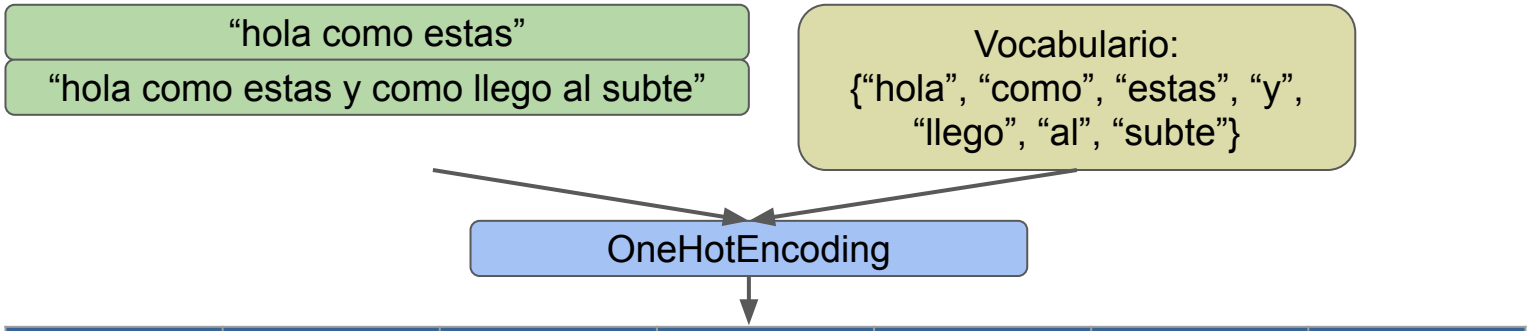
que	bueno	esta	muy	excelente
1	1	1	0	0
0	1	1	1	0
0	0	1	0	1

One-hot encoding (ejemplo)



hola	como	estas	y	llego	al	subte

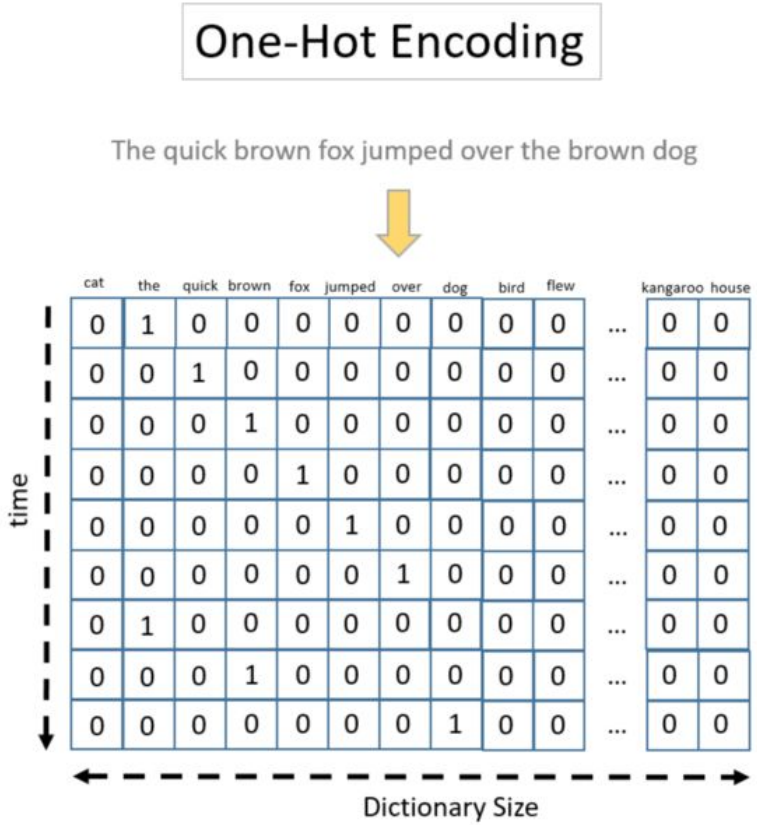
One-hot encoding (ejemplo resuelto)



hola	como	estas	y	llego	al	subte
1	1	1	0	0	0	0
1	1	1	1	1	1	1

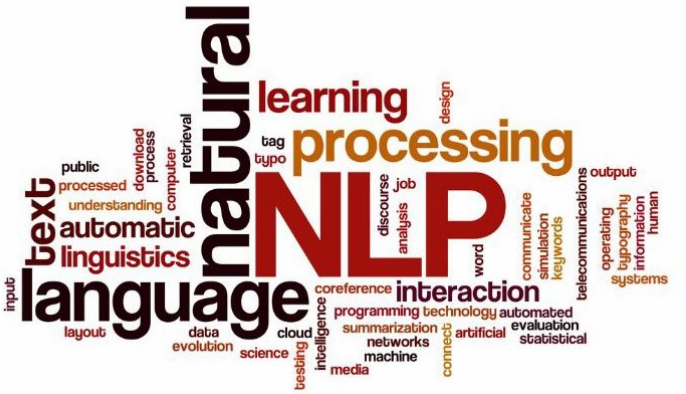
Los vectores tienen el largo del vocabulario

One-hot encoding



¡El idioma inglés tiene más de 180.000 palabras en su vocabulario en uso!

Bolsa de palabras "Bag of words" (BOW)



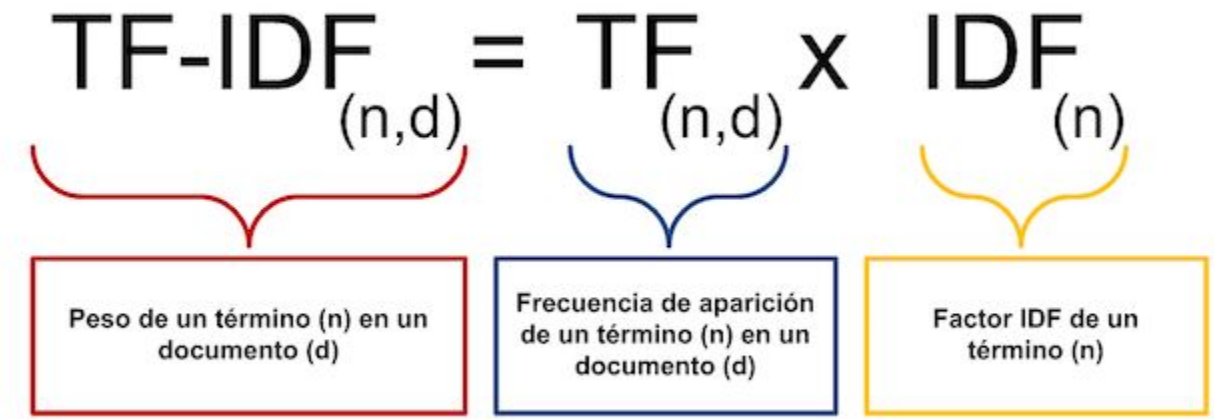
Representar a las palabras por su presencia o ausencia en el texto (y a veces la cantidad). Previo a la existencia de los embeddings y no tiene en consideración el contexto.

El problema es que los vectores de frecuencia o One-Hot encoding son muy “sparse”

*“Necesito **mucho espacio** para **guardar información** que **no aporta valor**”*

TF-IDF (Term frequency-Inverse term frequency)

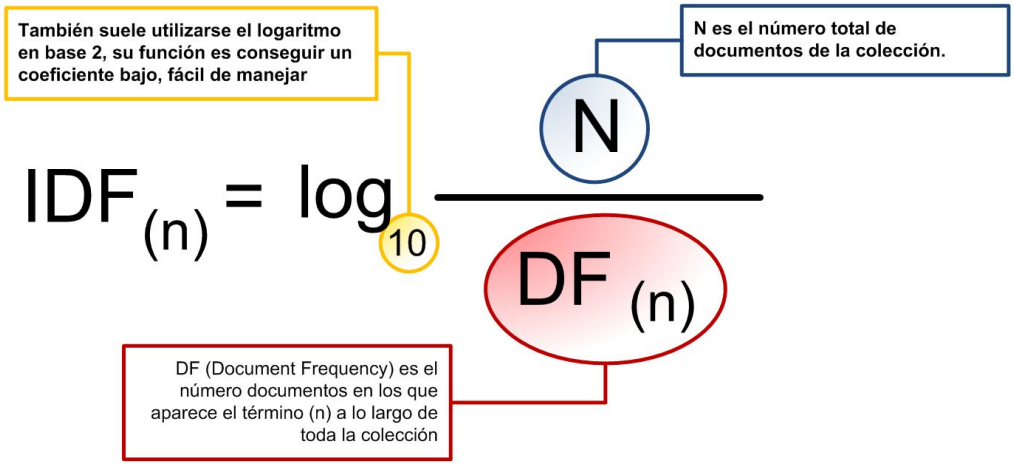
"Se utiliza como indicador de cuán importante es una palabra (término) en un documento"



El motor tan utilizado “Elasticsearch” se basa en este mecanismo

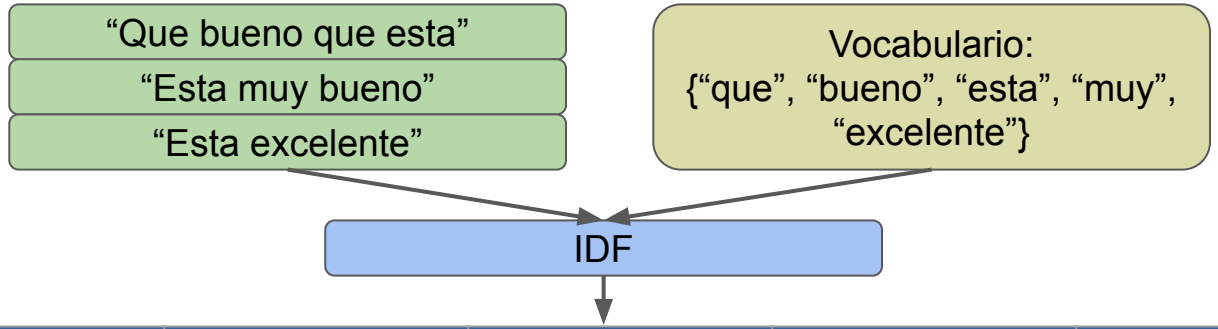
Factor IDF (Inverse Document Frequency)

"Proporción de documentos en el corpus que poseen el término"



Si el término aparece en todos los documentos el IDF será cero (es popular y por lo tanto aporta poco valor)

Factor IDF



que	bueno	esta	muy	excelente
log(3/1)	log(3/2)	log(3/3)	log(3/1)	log(3/1)

que	bueno	esta	muy	excelente
0.477	0.176	0	0.477	0.477

Se obtiene como la división de la cantidad de documentos sobre la suma en axis=0 (vertical) del OneHotEncoding.

Factor TF (Term frequency)

"Frecuencia de aparición de un término a lo largo de un documento"

$$tf(n) = \sum_{(n)}^{D1}$$

La frecuencia de aparición de un término (n) en un documento (D1) es la suma de las ocurrencias de dicho término

Se obtiene igual que el vector de frecuencia

Factor TF-IDF

"Que bueno que esta"
"Esta muy bueno"
"Esta excelente"

Vocabulario:
{"que", "bueno", "esta", "muy", "excelente"}

IDF

que	bueno	esta	muy	excelente
log(3/1)	log(3/2)	log(3/3)	log(3/1)	log(3/1)

TF-IDF

que	bueno	esta	muy	excelente
2 * log(3/1)	1 * log(3/2)	1 * log(3/3)	0 * log(3/1)	0 * log(3/1)
0 * log(3/1)	1 * log(3/2)	1 * log(3/3)	1 * log(3/1)	0 * log(3/1)
0 * log(3/1)	0 * log(3/2)	1 * log(3/3)	0 * log(3/1)	1 * log(3/1)

TF-IDF (ejemplo)

"hola como estas"

"hola como estas y como llego al subte"

Vocabulario:
??

TF

hola	como	estas	y	llego	al	subte

IDF

hola	como	estas	y	llego	al	subte

TF-IDF

hola	como	estas	y	llego	al	subte

TF-IDF (ejemplo resuelto)

"hola como estas"

"hola como estas y como llego al subte"

Vocabulario:
{"que", "bueno", "esta", "muy", "excelente"}

TF

hola	como	estas	y	llego	al	subte
1	1	1	0	0	0	0
1	2	1	1	1	1	1

IDF

hola	como	estas	y	llego	al	subte
log(2/2)	log(2/2)	log(2/2)	log(2/1)	log(2/1)	log(2/1)	log(2/1)

TF-IDF

hola	como	estas	y	llego	al	subte
0	0	0	0	0	0	0
0	0	0	log(2/1)	log(2/1)	log(2/1)	log(2/1)

Similitud coseno

"Se utiliza para evaluar la dirección de dos vectores"

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

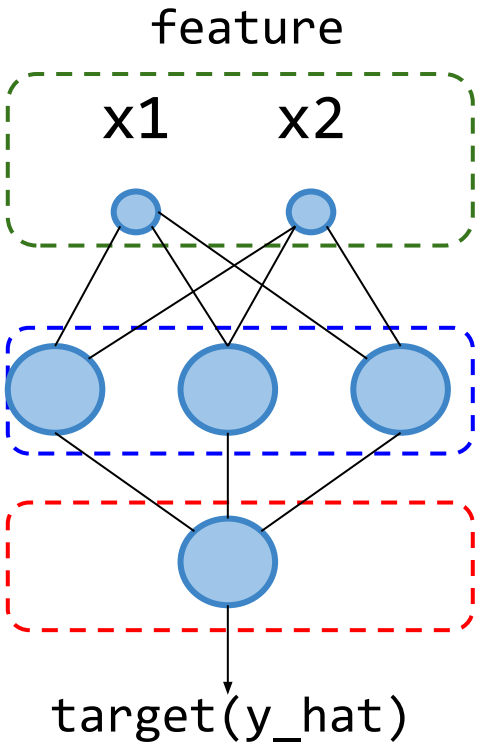
Similitud coseno = 1 → los vectores tienen la misma dirección.
Similitud coseno = 0 → los vectores son ortogonales.
Similitud coseno = -1 → los vectores apuntan en sentido contrario.

Desafio

Link al Colab

LINK

Keras & Tensorflow



input

Hidden

Output

```
# Crear un modelo secuencial
model = Sequential()

# Crear la capa de entrada y la capa oculta (hidden):
# --> tantas entradas (input_shape) como columnas de entrada
# --> tantas neuronas (units) como deseemos
# --> utilizamos "sigmoid" como capa de activación
model.add(Dense(units=3, activation='relu', input_shape=(2,)))

# Crear la output, tendrá tantas neuronas como salidas deseadas
model.add(Dense(units=1, activation='sigmoid'))
```

Keras Clasificación

Link al Colab

LINK

¡Muchas gracias!