

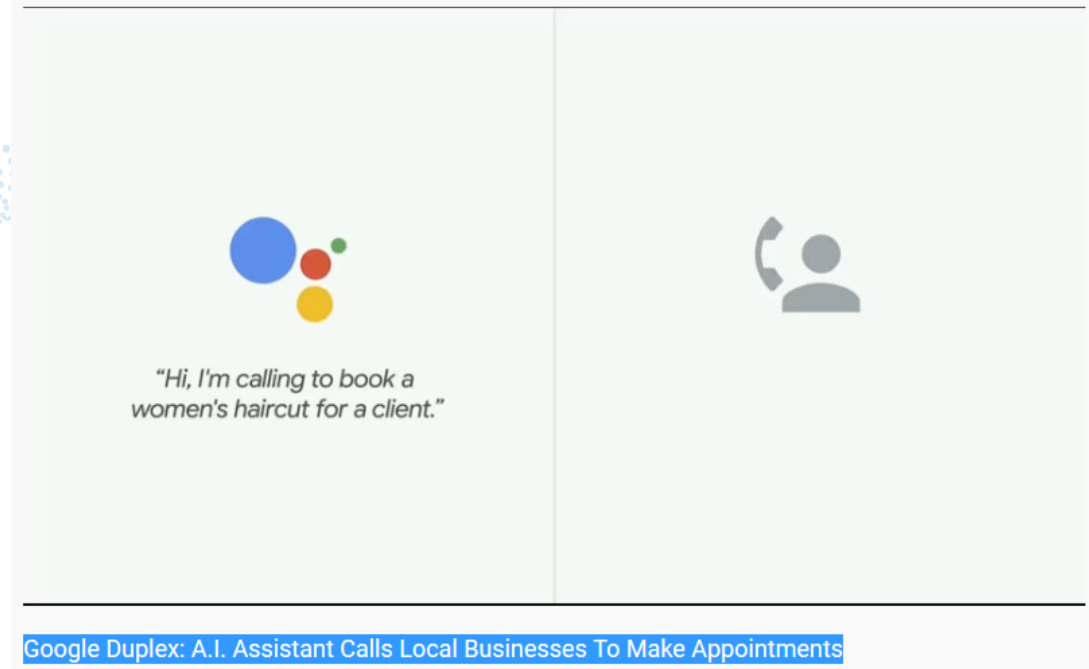
Procesamiento de Lenguaje Natural

A decorative graphic consisting of several parallel, wavy lines of blue dots. The dots are arranged in a way that creates a sense of motion and depth, flowing from the bottom left towards the top right of the frame. The background is a solid dark blue.

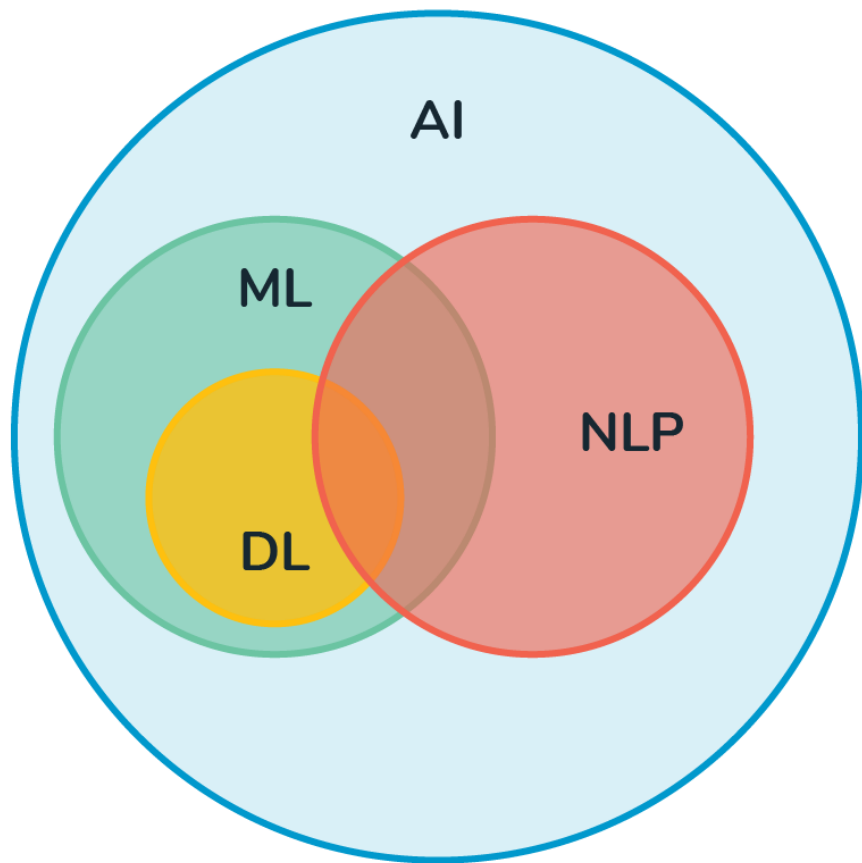
Contenido





- **Conceptos de Inteligencia Artificial / Machine Learning**
- **Conceptos de Procesamiento de Lenguaje Natural**
- **Conceptos de Análisis de Sentimientos**

Google Duplex: A.I. Assistant Calls Local Businesses To Make Appointments

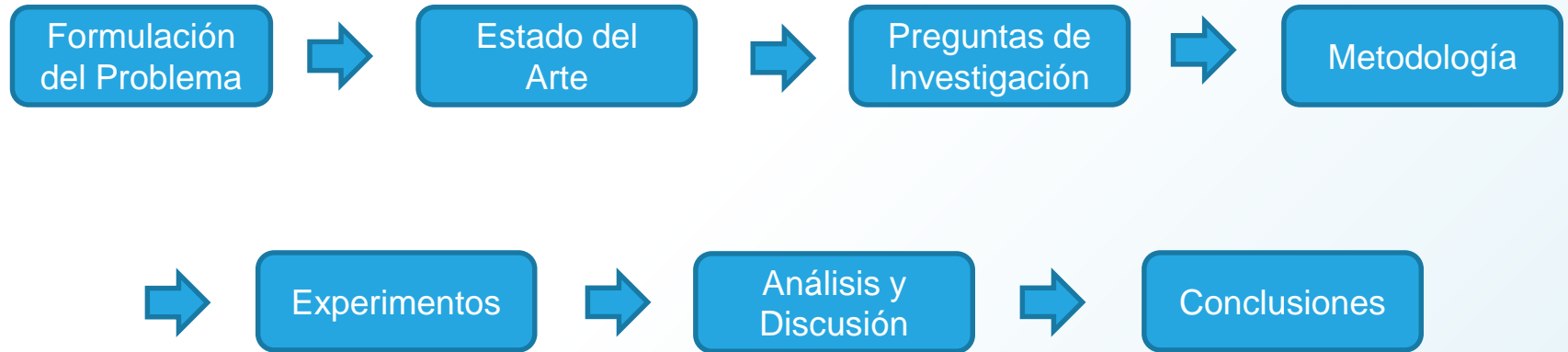


https://www.youtube.com/watch?v=D5VN56jQMWM&feature=emb_title

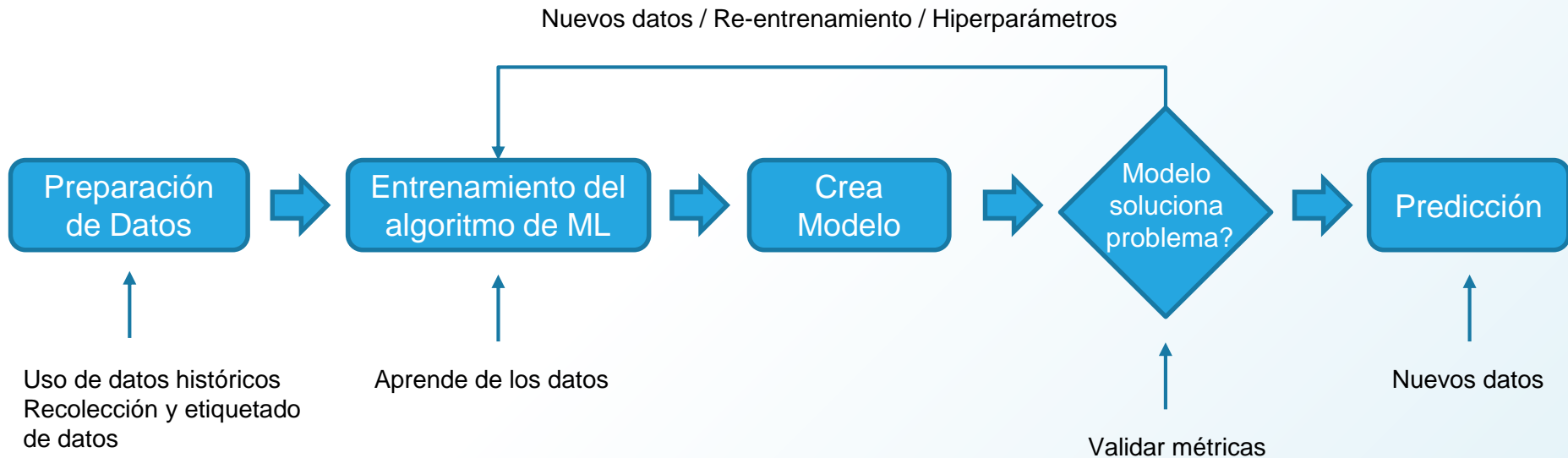


-  Artificial intelligence
-  Machine learning
-  Language Processing
-  Deep learning

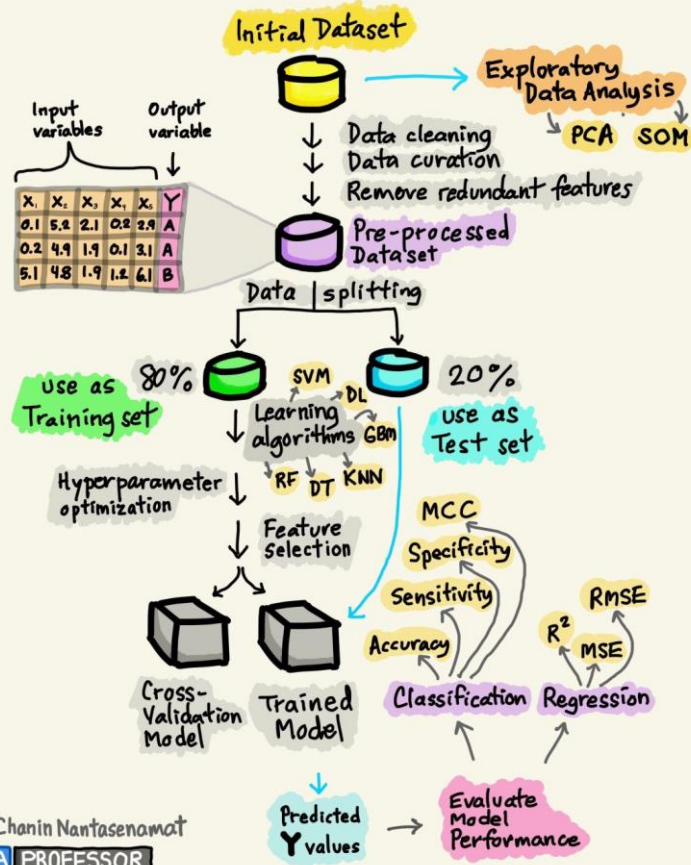
Ciclo de Vida – Trabajo de Investigación



Ciclo de Vida – Experimentos en Machine Learning



BUILDING THE MACHINE LEARNING MODEL



By: Chanin Nantasenamat

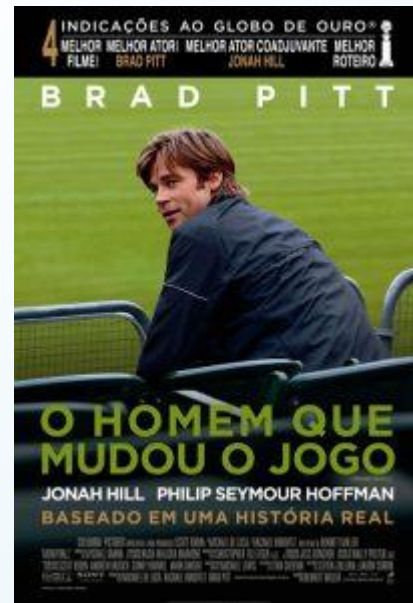
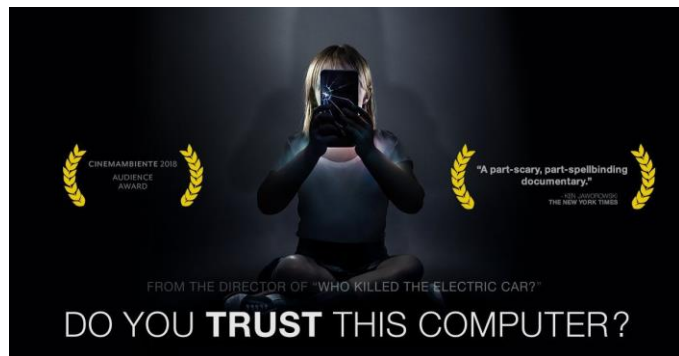
DATA PROFESSOR



<http://youtube.com/dataprofessor>

January 1, 2020

Videos

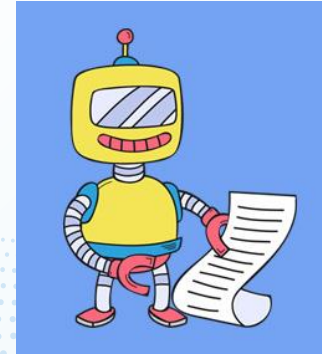


1. Definición

Qué es Procesamiento de Lenguaje Natural?

Lenguaje Natural

- Comunicación por seres humanos
- Ejemplos: Inglés, Español, Portugués
- Procesamiento de Lenguaje Natural es cualquier manipulación computacional del lenguaje natural



Tecnologías basadas en PLN

- Predicción de texto / Correctores Ortográficos (MS Word/otros editores de texto)
- Búsqueda en la web (Google, Bing, Yahoo)
- Clasificadores de Spam (Todos los servicios de email)
- Traducción automática (Google Translate)
- Asistentes virtuales (Siri, Alexa, Cortana)
- Análisis de sentimientos en tweets y blogs

Técnicas

- Segmentación de Oraciones
- Segmentación de Palabras (Tokenización)
- Limpieza de datos (stopwords)
- Análisis Lexicográfico (stemming, lematizing)
- Etiquetado Gramatical (POS)
- Chunking
- Reconocimiento de Entidades

NLTK

- Instalación:

```
import nltk  
nltk.download('book')
```

- Carga de textos:

```
from nltk.book import *  
text9
```

```
<Text: The Man Who Was Thursday by G . K . Chesterton 1908>
```

Contador de Vocabulario

```
print('Cantidad de símbolos:',len(text3))
```

Cantidad de símbolos: 44764

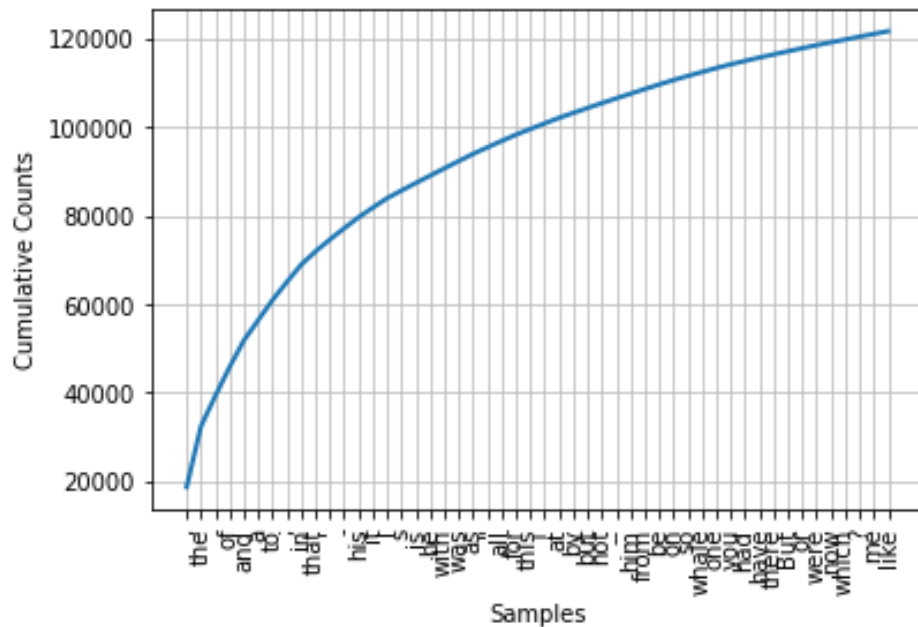
```
print('Cantidad de palabras',len(set(text3)))
```

Cantidad de palabras 2789

```
text3.count("smote")
```

Distribución de Frecuencia

```
fdist = FreqDist(text1)
fdist.most_common(50)
fdist.plot(50, cumulative=True)
```



Tokenización

it not cool that ping pong is not included in rio 2016



Tokenization

it

not

cool

that

ping

pong

is

not

included

in

rio

2016

Tokenization

```
text = word_tokenize("And now for something completely different")  
print(nltk.pos_tag(text))
```

```
Tokens: ['And', 'now', 'for', 'something', 'completely', 'different']
```

Stopwords y Signos de Puntuación



Stop Words: remover palabras comunes pero que no proveen utilidad al descubrimiento del contexto (el, la, de, los, y, etc...)

StopWords

```
from nltk.corpus import stopwords

def removeStopword(texto):
    stopwordSpanish = stopwords.words('spanish')
    textNew= [w.lower() for w in texto if w not in stopwordSpanish]
    print(textNew)
```

```
['A', 'punto', 'de', 'cumplirse', 'una', 'semana', 'desde', 'que', 'Nicolás', 'Maduro', 'pusiera',  
'en', 'marcha', 'las', 'nuevas', 'medidas', 'económicas', 'para', 'Venezuela', ',']  
['a', 'punto', 'cumplirse', 'semana', 'nicolás', 'maduro', 'pusiera', 'marcha', 'nuevas',  
'medidas', 'económicas', 'venezuela', ',,', 'valor', 'bolívar', ',,', 'fijado', 'tasa', '60',  
'unidades']
```

Stemming - Lemmatization

campo^o --> camp

casita --> cas

vendedor --> vend

panadería --> pan

comiendo --> comer

limones --> limón

corruptas --> corruptos

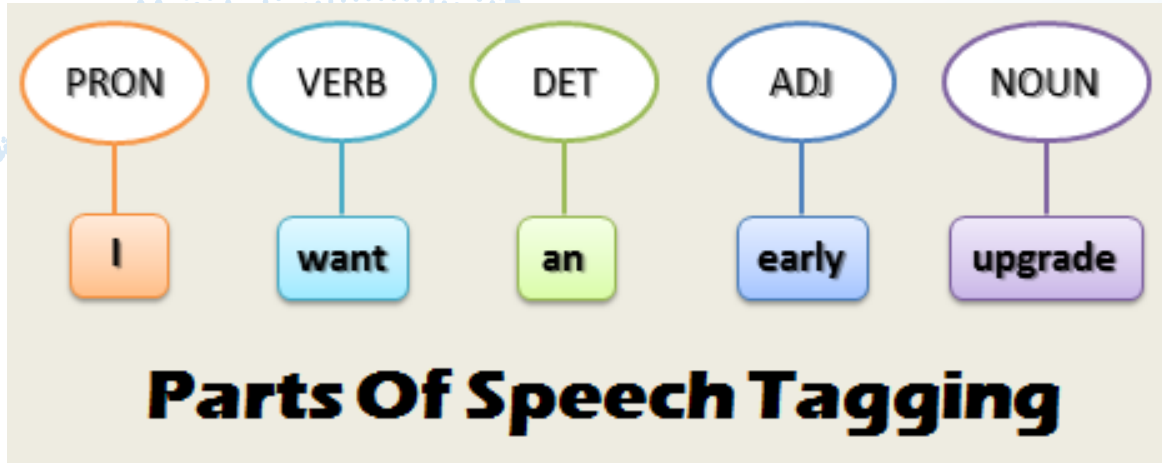
nueces --> nuez

Stemming y Lemmatization

```
tokens = word_tokenize(raw)
porter = nltk.PorterStemmer()
lancaster = nltk.LancasterStemmer()
print('PorterStemmer',[porter.stem(t) for t in tokens])
print('LancasterStemmer',[lancaster.stem(t) for t in tokens])
wnl = nltk.WordNetLemmatizer()
print('WordNetLemmatizer',[wnl.lemmatize(t) for t in tokens])
```

```
Porter Stemmer ['la', 'familia', 'suelen', 'escond', 'a', 'lo', 'niño', 'afectado', 'o', 'lo', 'aíslan', 'con', 'lo', 'animal']
Lancaster Stemmer ['las', 'familia', 'suel', 'escond', 'a', 'los', 'niño', 'afectado', 'o', 'los', 'aísl', 'con', 'los', 'anim']
Lemmatization ['Las', 'familias', 'suelen', 'esconder', 'a', 'los', 'niños', 'afectados', 'o', 'los', 'aíslan', 'con', 'los', 'animales']
```

Etiquetado Gramatical Tagging Words



Etiquetador (tagger)

```
text = word_tokenize("And now for something completely different")  
nltk.pos_tag(text)
```

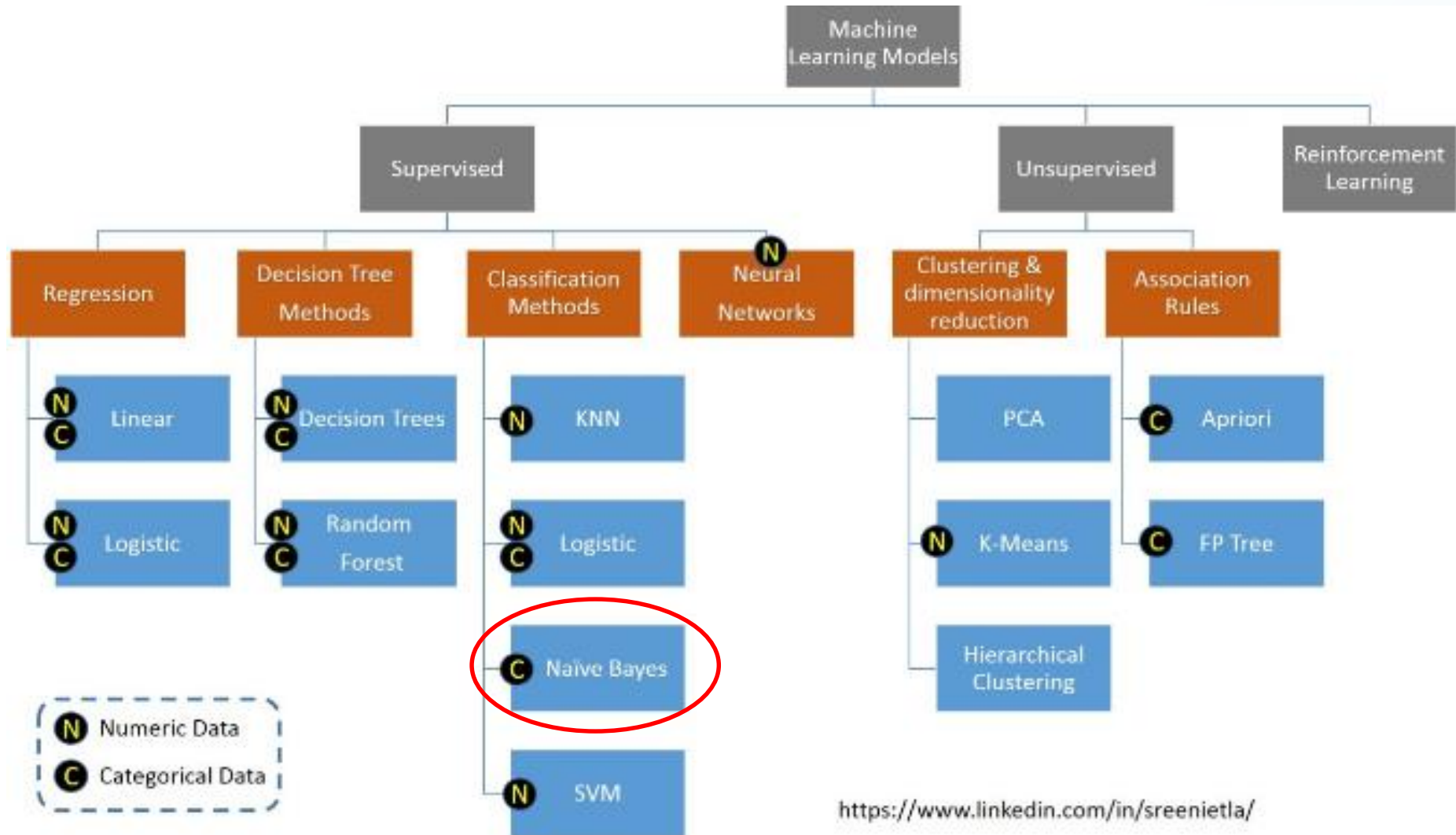
```
[('And', 'CC'),  
 ('now', 'RB'),  
 ('for', 'IN'),  
 ('something', 'NN'),  
 ('completely', 'RB'),  
 ('different', 'JJ')]
```

```
nltk.corpus.brown.tagged_words(tagset='universal')
```

```
[('The', 'DET'), ('Fulton', 'NOUN'), ...]
```

2. Clasificación Supervisada

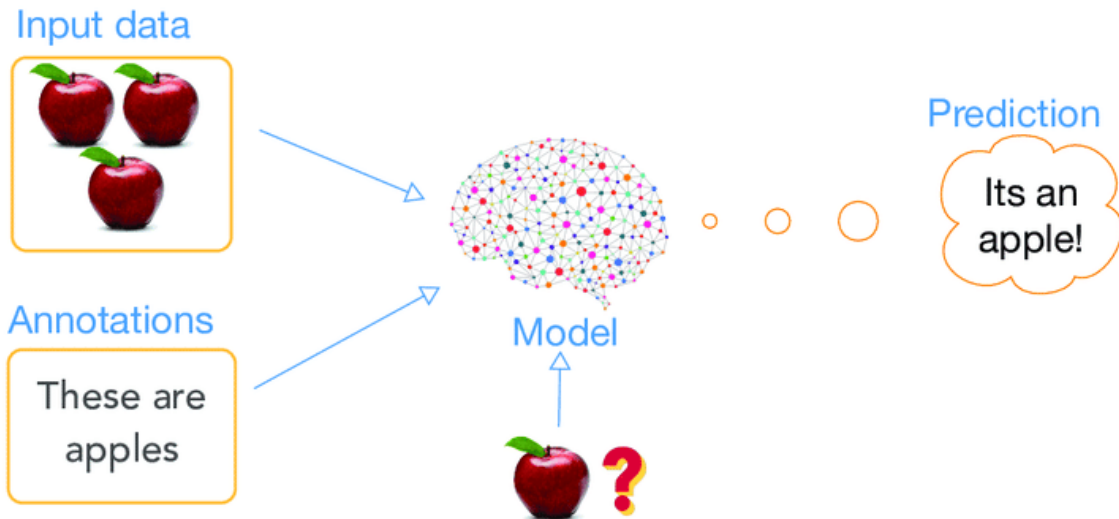
Y su uso para el Análisis de Sentimientos



Aprendizaje Automático

Clasificación Supervisada

supervised learning

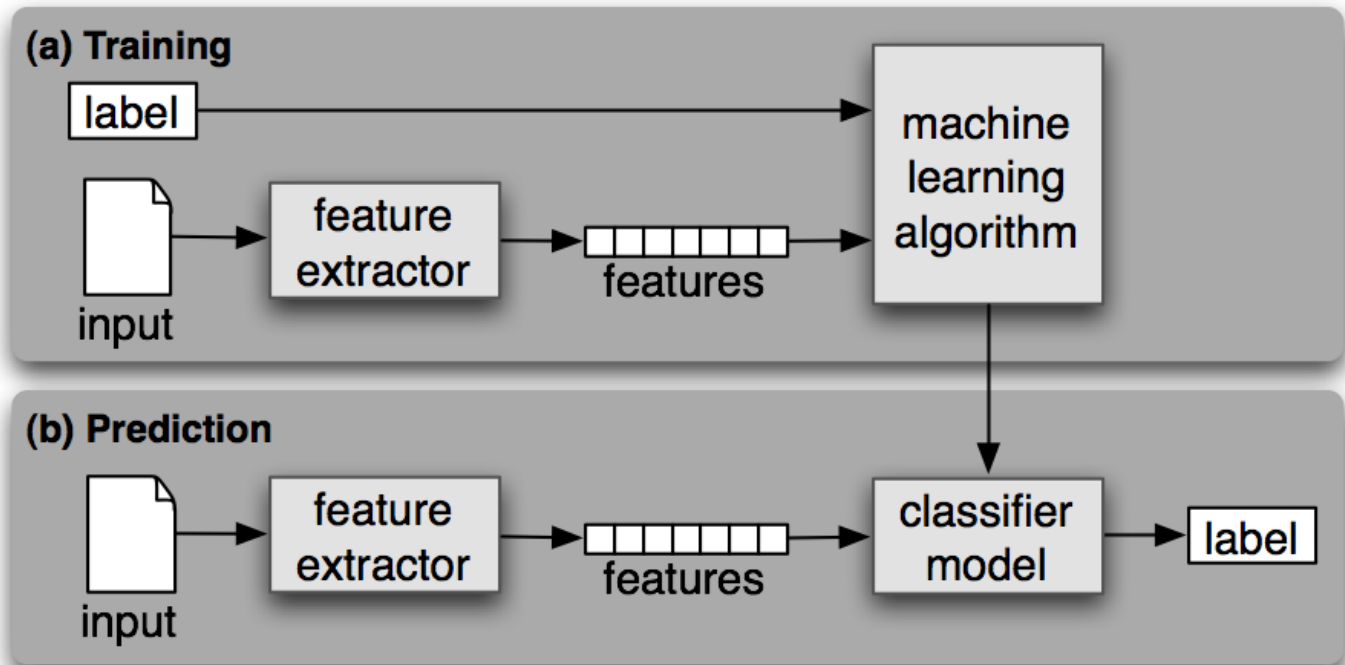


Análisis de Sentimientos



Aprendizaje Automático

Clasificación Supervisada



Aprendizaje Automático

Clasificación Supervisada

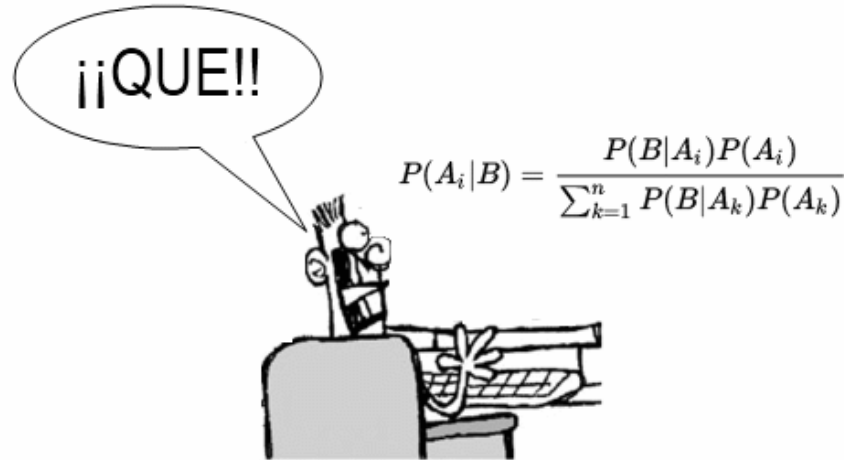
Características (features)



Técnicas

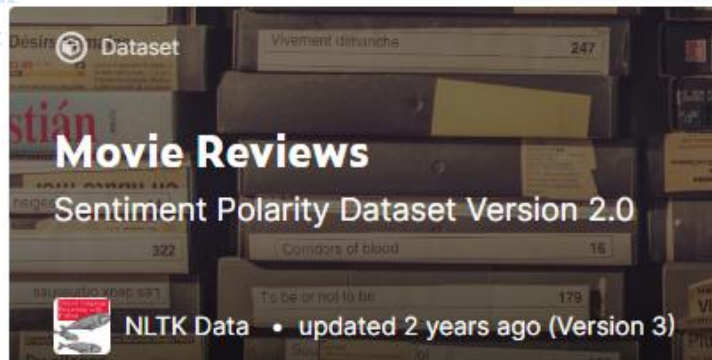
- Tokenización
- Lematización
- POS
- NER

Naive Bayes



Movie Reviews

- 1000 archivos etiquetados como positivos y otros 1000 etiquetados como negativos.



Movie Reviews

```
def Estadisticas():
```

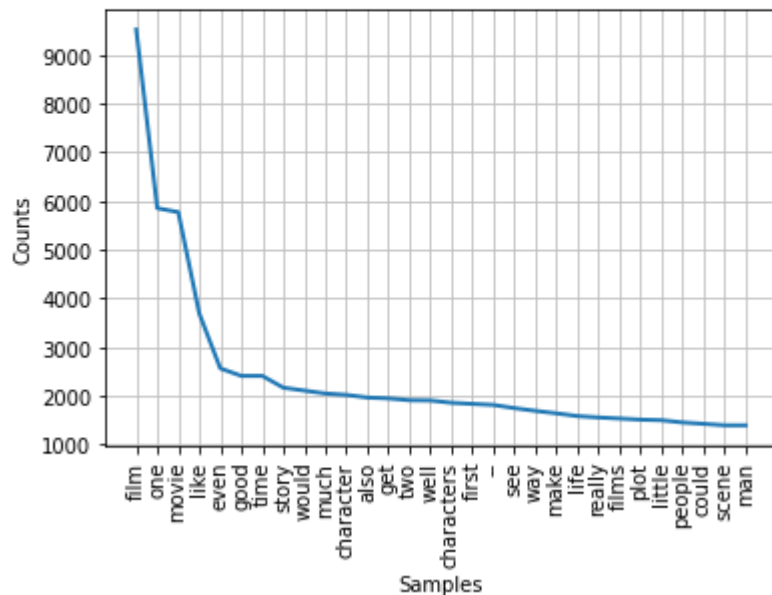
```
    print ('total',len(movie_reviews.fileids()))
    print ('categorias',movie_reviews.categories())
    print ('total positivos',len(movie_reviews.fileids('pos')))
    print ('total negativos',len(movie_reviews.fileids('neg')))

    all_words = [word.lower() for word in movie_reviews.words()]
    all_words_frequency = FreqDist(all_words)
    print ('10 palabras más frecuentes',all_words_frequency.most_common(10))
    print ('cantidad de veces que se repite la palabra happy',all_words_frequency['happy'])
```

```
total 2000
categorias ['neg', 'pos']
total positivos 1000
total negativos 1000
10 palabras más frecuentes [(' ', 77717), ('the', 76529), ('.', 65876), ('a', 38106),
('and', 35576), ('of', 34123), ('to', 31937), ('"', 30585), ('is', 25195), ('in', 21822)]
cantidad de veces que se repite la palabra happy 215
```


Movie Reviews

total positivos 1000
total negativos 1000
1583820
710578
10 palabras más frecuentes [('film', 9517), ('one', 5852)].
cantidad de veces que se repite la palabra happy 215



Bag of Words Feature

- **Bag of Words** method converts text data into numbers.
- It does this by
 - Creating a **vocabulary** from the words in all documents
 - Calculating the **occurrences** of words:
 - **binary** (present or not)
 - **word counts**
 - **frequencies**

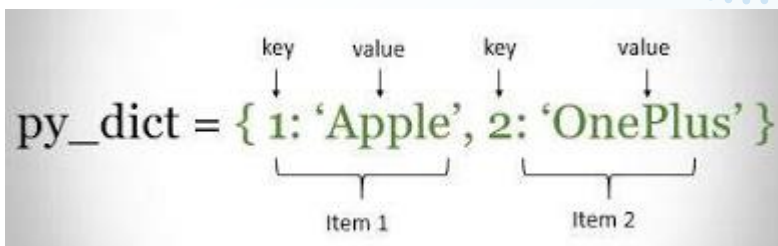
Bag of Words Feature

Simple example using word counts:

	a	cat	dog	happy	is	it	my	not	old	wolf
"It is a dog."	1	0	1	0	1	1	0	0	0	0
"my cat is old"	0	1	0	0	1	0	1	0	1	0
"It is not a dog, it a is wolf."	2	0	1	0	2	2	0	1	0	1

Bag of Words Feature

```
def bag_of_words(words):  
    words_clean = []  
    stopwords_english = stopwords.words('english')  
  
    for word in words:  
        word = word.lower()  
        if word not in stopwords_english and word not in string.punctuation:  
            words_clean.append(word)  
  
    words_dictionary = dict([word, True] for word in words_clean)  
  
    return words_dictionary
```



Bag of Words Feature

```
def DatosBOW():  
    pos_reviews = []  
    for fileid in movie_reviews.fileids('pos'):  
        words = movie_reviews.words(fileid)  
        pos_reviews.append(words)  
  
    neg_reviews = []  
    for fileid in movie_reviews.fileids('neg'):  
        words = movie_reviews.words(fileid)  
        neg_reviews.append(words)  
  
    pos_reviews_set = []  
    for words in pos_reviews:  
        pos_reviews_set.append((bag_of_words(words), 'pos'))  
  
    neg_reviews_set = []  
    for words in neg_reviews:  
        neg_reviews_set.append((bag_of_words(words), 'neg'))  
  
    return pos_reviews_set, neg_reviews_set
```

```
POSITIVO [({'plot': True, 'two':  
True, 'teen': True, 'couples':  
True, 'go': True, 'church': True,  
'party': True, 'drink': True,  
'drive': True, 'get': True,  
'accident': True, 'one': True,  
'guys': True, 'dies': True,  
'girlfriend': True, 'continues':  
True, 'see': True, 'life': True,  
'nightmares': True, 'deal': True,  
'watch': True, 'movie': True,  
'sorta': True, 'find': True,  
'critique': True, 'mind': True,  
'fuck': True, 'generation': True,  
'touches': True, 'cool': True,  
'idea': True, 'presents': True,
```

```
NEGATIVO [({'plot': True, 'two':  
True, 'teen': True, 'couples':  
True, 'go': True, 'church': True,  
'party': True, 'drink': True,  
'drive': True, 'get': True,  
'accident': True, 'one': True,  
'guys': True, 'dies': True,  
'girlfriend': True, 'continues':  
True, 'see': True, 'life': True,  
'nightmares': True, 'deal': True,  
'watch': True, 'movie': True,  
'sorta': True, 'find': True,  
'critique': True, 'mind': True,  
'fuck': True, 'generation': True,
```

```
def clasificadorBOW(pos_reviews_set,neg_reviews_set):  
    size = int(len(pos_reviews_set) * 0.1)  
    test_set = pos_reviews_set[:size] + neg_reviews_set[:size]  
    train_set = pos_reviews_set[size:] + neg_reviews_set[size:]  
    print(len(test_set), len(train_set))  
    classifier = NaiveBayesClassifier.train(train_set)  
    accuracy = classify.accuracy(classifier, test_set)  
    print(accuracy)  
    print (classifier.show_most_informative_features(10))  
    return classifier
```

```
def pruebaBOW(custom_review,classifier):  
    custom_review_tokens = word_tokenize(custom_review)  
    custom_review_set = bag_of_words(custom_review_tokens)  
    print (custom_review)  
    print ('positivo o negativo?',classifier.classify(custom_review_set))  
    prob_result = classifier.prob_classify(custom_review_set)  
    print ('probabilidad para negativo',prob_result.prob("neg"))  
    print ('probabilidad para positivo',prob_result.prob("pos"))
```

I hated the film. It was a disaster. Poor direction, bad acting.

positivo o negativo? neg

probabilidad para negativo 0.8389515850310557

probabilidad para positivo 0.16104841496894456

It was a wonderful and amazing movie. I loved it. Best direction, good acting.

positivo o negativo? pos

probabilidad para negativo 0.06519158529235963

probabilidad para positivo 0.9348084147076396

Matriz de Confusión

		Realidad	
		Positivos	Negativos
Predicción	Positivos	TP	FP
	Negativos	FN	TN

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN} \quad (\text{sensitivity})$$

$$F - score = \frac{2 \times (Precision \times Recall)}{(Precision + Recall)}$$

$$Specificity = \frac{TN}{TN + FP}$$

4. Herramientas Actuales

Qué puedo usar?

Herramientas actuales

ARTIFICIAL INTELLIGENCE

Amazon Transcribe

Automatic Speech Recognition (ASR) powered by deep learning

Amazon Transcribe provides quality and affordable speech recognition for efficient speech to text transcription.

Watson Natural Language Understanding

The natural language processing (NLP) service for advanced text analytics

[Get started free](#)

[View demo](#)



Herramientas actuales

Google launches AutoML Natural Language with improved text classification and model training

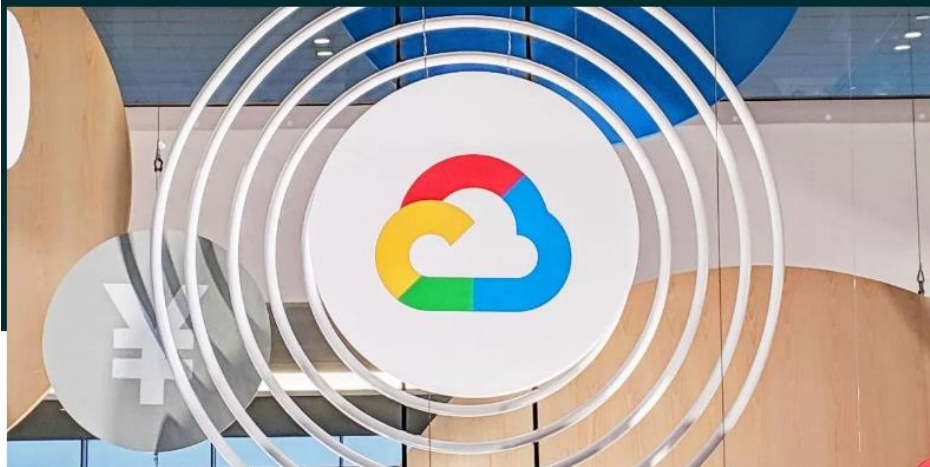
Kyle Wiggers

@Kyle_L_Wiggers

December 12, 2019 9:56 AM

AI

f t in



Herramientas actuales



Natural Language Analysis
with Python NLTK

Deep Learning NLP with spaCy

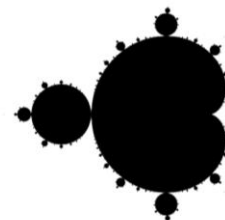
spaCy

Herramientas actuales



OPEN NLP™

Herramientas actuales



For NLP Preprocessing

TextBlob

NLP  ARCHITECT

The logo for NLP ARCHITECT features the text 'NLP' in a large, black, sans-serif font, followed by an orange isometric wireframe cube icon, and then the word 'ARCHITECT' in a large, black, sans-serif font.

Gracias!

Alguna pregunta?

- Maria.limaylla@gmail.com