

Clase 2: Pre-procesamiento de textos

Enfoques Clásicos y Neuronales a la Minería de Texto

Marcelo Errecalde^{1,2}

¹Universidad Nacional de San Luis, Argentina 

²Universidad Nacional de la Patagonia Austral, Argentina 



Resumen

1 KDD a partir de textos

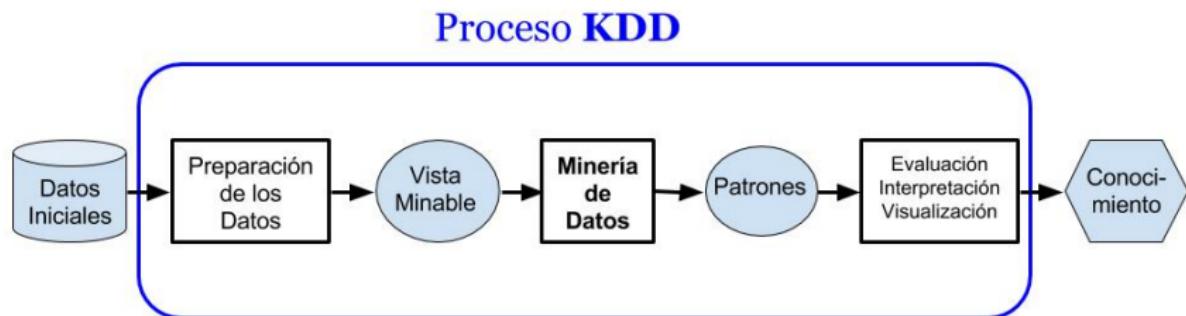
2 Pre-procesamiento

- Partición del texto
- Filtrado de palabras
- Normalización de palabras
- Etiquetado de palabras

KDD a partir de textos

Sirve de guía para la organización de los contenidos de este curso.

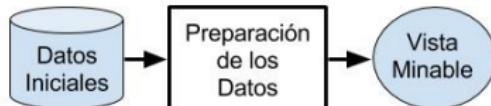
Fases del Proceso KDD



Fase de Preparación de los textos

Sirve de guía para la organización de los contenidos de este curso.

Fases del Proceso KDD



Fase de Preparación de los textos

- Recopilación de textos
 - Pre-procesamiento de los textos
 - Representación de los textos
 - Reducción de dimensionalidad

Algunas técnicas de pre-procesamiento

- 1 Partición del texto
 - 2 Filtrado (“stop-words”, baja frecuencia)
 - 3 Normalización (mayúsculas, variaciones de uso)
 - 4 Truncado (“stemming”) y lematización (“lemmatization”)
 - 5 Etiquetado de las palabras
 - De Partes de la Oración (Part of Speech (**POS**) Tagging)
 - Desambiguación del Significado de las Palabras (**WSD**)
 - Reconocimiento de Entidades Nombradas (**NER**)

Tokenización

Ejemplo, dada la siguiente sentencia:

After sleeping for four hours, he decided to sleep for another four.

El proceso de tokenización produciría:

{ “After” “sleeping” “for” “four” “hours” “he” “decided”
“to” “sleep” “for” “another” “four”}.

Comparación truncado vs lematización

In [1]:

```
compare_normalization(u"Our meeting today was worse than yesterday, "
                      "I'm scared of meeting the clients tomorrow.")
```

Comparación truncado vs lematización

In [1]:

```
compare_normalization(u"Our meeting today was worse than yesterday,  
"I'm scared of meeting the clients tomorrow.")
```

Out[1]:

Lemmatization:

```
['our', 'meeting', 'today', 'be', 'bad', 'than', 'yesterday', ',',  
'i', 'be', 'scared', 'of', 'meet', 'the', 'client', 'tomorrow', '.']
```

Stemming:

```
['our', 'meet', 'today', 'wa', 'wors', 'than', 'yesterday', ',',  
'i', "'m", 'scare', 'of', 'meet', 'the', 'client', 'tomorrow', '.']
```

Etiquetado de las Categorías Gramaticales

Ejemplo: las palabras de la sentencia

The grand jury commented on a number of other topics.

Etiquetado de las Categorías Gramaticales

Ejemplo: las palabras de la sentencia

The grand jury commented on a number of other topics.

podrían ser etiquetadas con las siguientes categorías por un sistema de ECG:

The/**DT** grand/**JJ** jury/**NN** commented/**VBD** on/**IN** a/**DT** number/**NN** of/**IN** other/**JJ** topics/**NNS** ./.

KDD a partir de textos

oooooooooooo●oooooooooooooooooooooooooooo

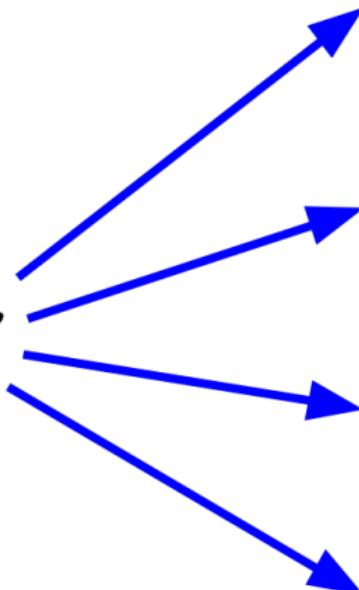
Pre-procesamiento

oooooooooooooooooooooooooooooooooooo

¿Qué significa “banco”?

¿Qué significa “banco”?

“banco”



Desambiguación del Significado de las Palabras

Definición

En inglés **Word Sense Disambiguation (WSD)** trata de resolver la ambigüedad en el significado de las palabras (o frases) en base al contexto en que éstas aparecen.

Ejemplo, la palabra banco...

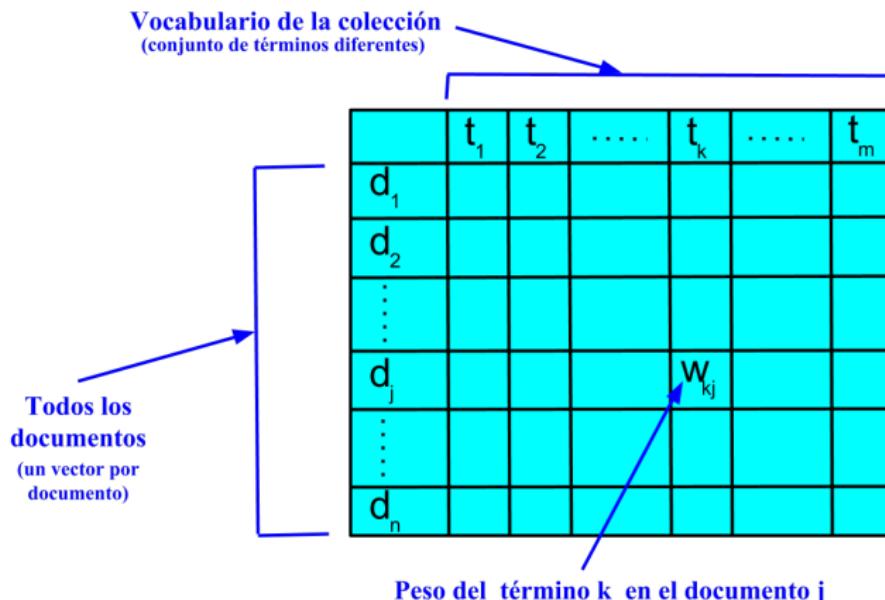
Frase		Significado
Perdí la mañana en el banco pagando impuestos.	⇒	Institución Financiera
Desde el barco vi el banco de peces.	⇒	Cardumen
Sentado en un banco suspiraba.	⇒	Mueble
Las donaciones se están recibiendo en el banco de sangre.	⇒	Establecimiento Médico

Reconocimiento de Entidades Nombradas

Características

- Subtarea de Extracción de Información.
- Consiste en la **ubicación** y **clasificación** de palabras dentro de un texto, en categorías tales como **nombres de personas, lugares, organizaciones, cantidades**, etc.
- El resultado de este proceso es un documento con etiquetas que identifican el comienzo y fin de las entidades nombradas.
- Ejemplo: “Jim bought 300 shares of Acme Corp. in 2006”
⇒ <ENAMEX TYPE=“PERSON”>Jim</ENAMEX> bought <NUMEX TYPE=“QUANTITY”>300</NUMEX> shares of <ENAMEX TYPE=“ORGANIZATION”>Acme Corp.</ENAMEX> in <TIMEX TYPE=“DATE”>2006</TIMEX>.

Representación vectorial de documentos: visión general



Representación de *bolsa de palabras* ("Bag of Words" (BoW))

Documentos

- ① "pintaron el banco de la plaza"
- ② "si paso la prueba, iremos paso a paso"
- ③ "no me banco ir al banco a cobrar cheques"

Representación BoW

D/T	a	al	banco	cheques	cobrar	de	el	ir	iremos	la	me	no	paso	pintaron	plaza	prueba	si
d1	0	0	1	0	0	1	1	0	0	1	0	0	0	1	1	0	0
d2	1	0	0	0	0	0	0	0	1	1	0	0	3	0	0	1	1
d3	1	1	2	1	1	0	0	1	0	0	1	1	0	0	0	0	0

Reducción de dimensionalidad: LSA

$$\mathbf{X} = \begin{matrix} & t_1 & t_2 & \dots & t_i & \dots & t_m \\ d_1 & & & & & & \\ d_2 & & & & & & \\ \vdots & & & & & & \\ d_l & & & & & & \\ \vdots & & & & & & \\ d_n & & & & & & \end{matrix}$$

$|D| \times |T|$

=

$$\mathbf{X} = \mathbf{U}_k \mathbf{S}_k \mathbf{V}_k^T$$

$$\mathbf{U}_k = \begin{matrix} & c_1 & c_2 & \dots & c_k \\ d_1 & & & & \\ d_2 & & & & \\ \vdots & & & & \\ d_l & & & & \\ \vdots & & & & \\ d_n & & & & \end{matrix}$$

$|D| \times k$

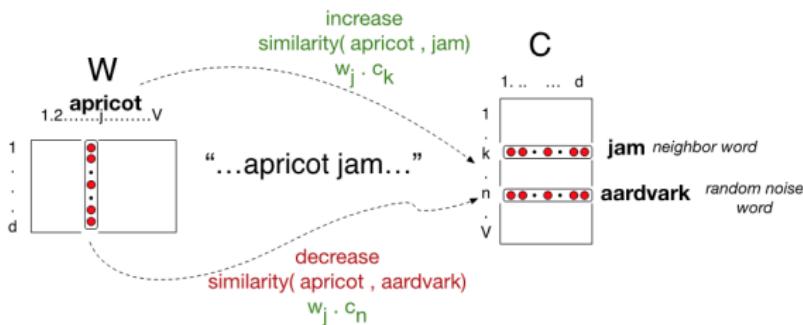
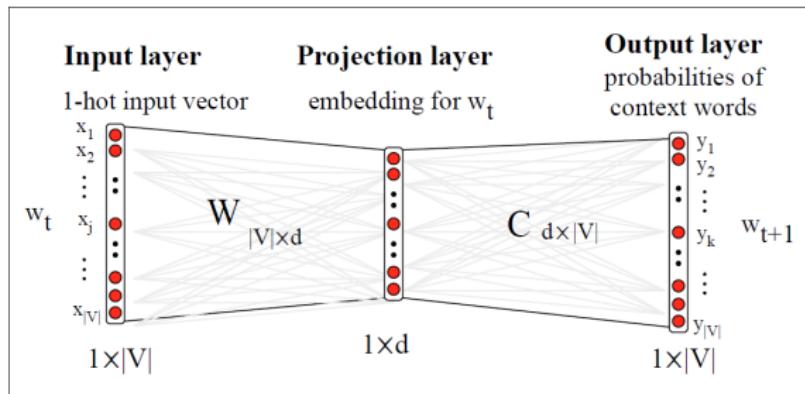
$$\mathbf{S}_k = \begin{matrix} & c_1 & c_2 & \dots & c_k \\ c_1 & & & & \\ c_2 & & & & \\ \vdots & & & & \\ c_k & & & & \end{matrix}$$

$k \times k$

$$\mathbf{V}_k^T = \begin{matrix} & t_1 & t_2 & \dots & t_i & \dots & t_m \\ c_1 & & & & & & \\ c_2 & & & & & & \\ \vdots & & & & & & \\ c_k & & & & & & \end{matrix}$$

$k \times |T|$

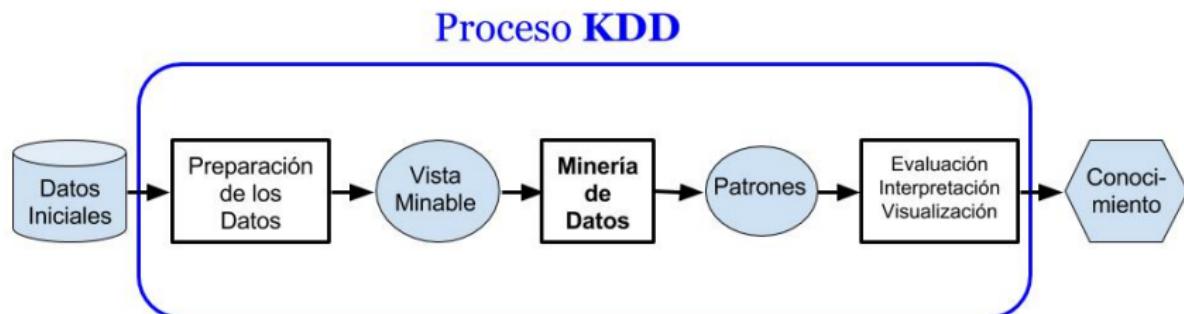
Reducción de dimensionalidad: word2vec



KDD a partir de textos

Sirve de guía para la organización de los contenidos de este curso.

Fases del Proceso KDD



Fase de Minería de textos

Sirve de guía para la organización de los contenidos de este curso.

Fases del Proceso KDD



Fase de Minería de textos

Involucra distintas **tareas de análisis**:

- **Categorización** de textos
- **Agrupamiento** de textos
- Modelización de **tópicos**
- **Extracción** de Información

Categorización de textos

Datos

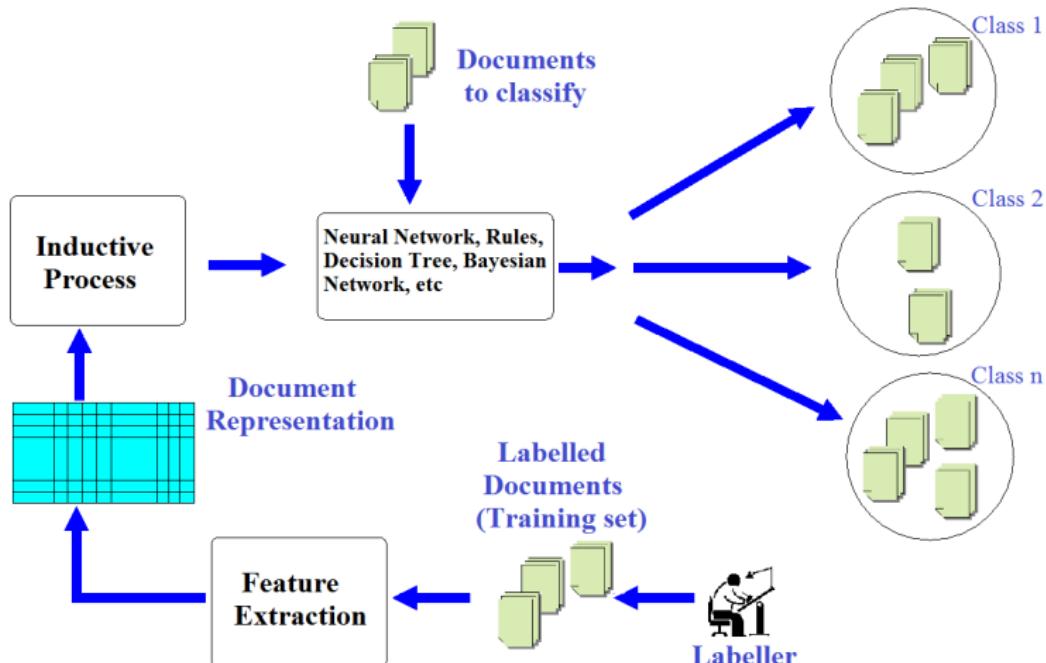
- Una colección de documentos \mathcal{D}
- Un conjunto de categorías $\mathcal{C} = \{c_1, \dots, c_{|\mathcal{C}|}\}$

Categorización de textos es la tarea de asignar los documentos en \mathcal{D} a las categorías en \mathcal{C}

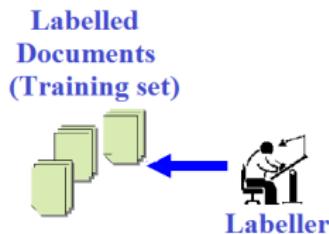
Ejemplos:

Problema	Objeto	Categorías (\mathcal{C})
Detección de spam	e-mails	$\{\text{True}, \text{False}\}$
Identificación de autores	documentos	autores
Categorización de noticias	cables de noticias	secciones del periódico
WSD	palabras con su contexto	significados de la palabra

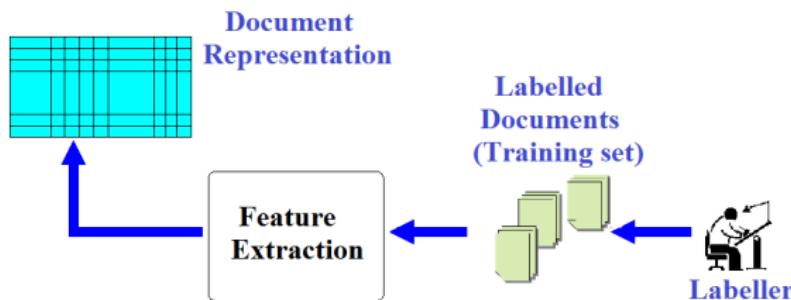
Ejemplo: Categorización Supervisada de Textos



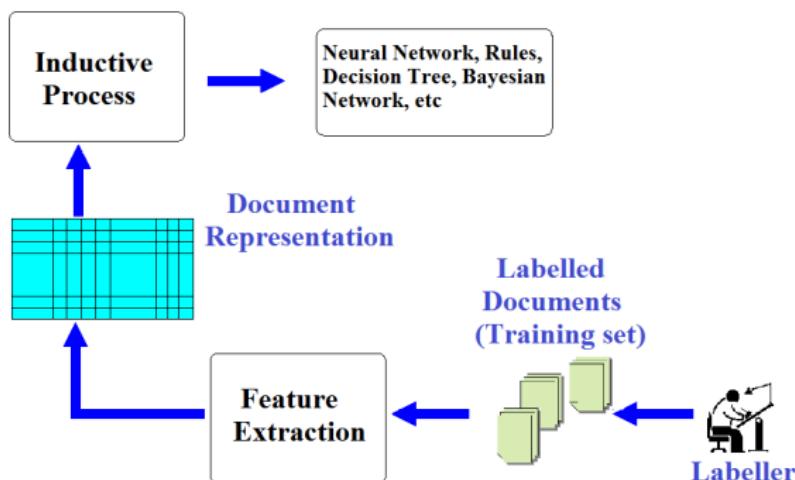
Ejemplo: Categorización Supervisada de Textos



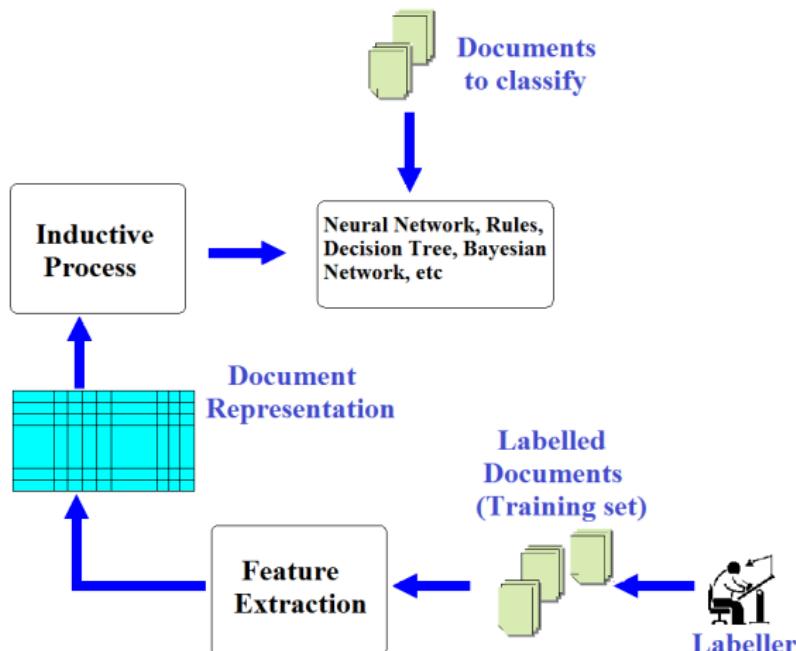
Ejemplo: Categorización Supervisada de Textos



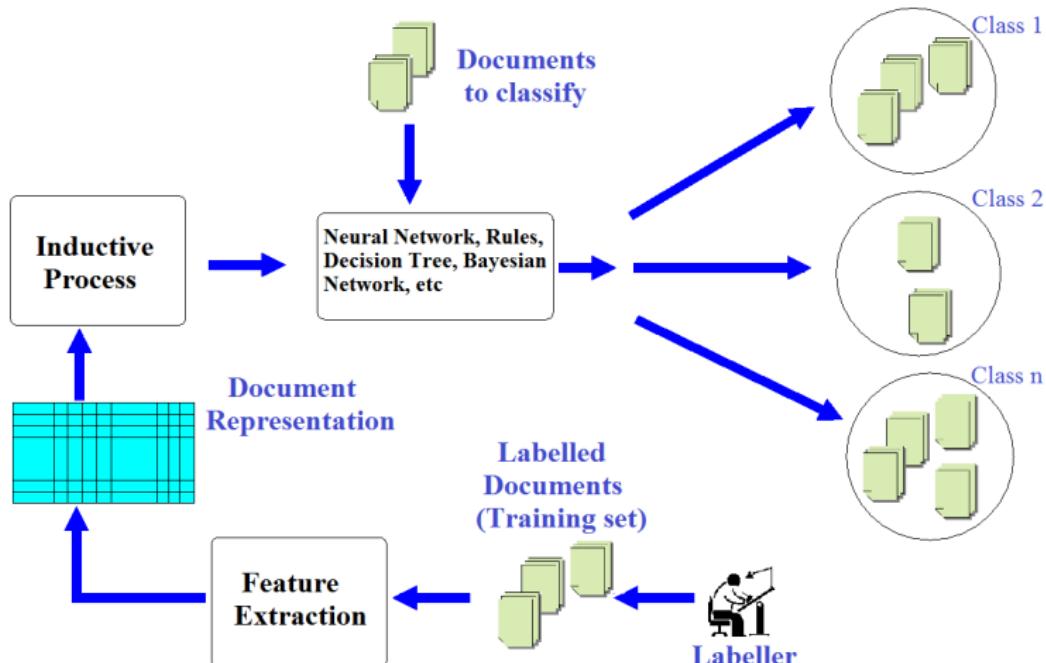
Ejemplo: Categorización Supervisada de Textos



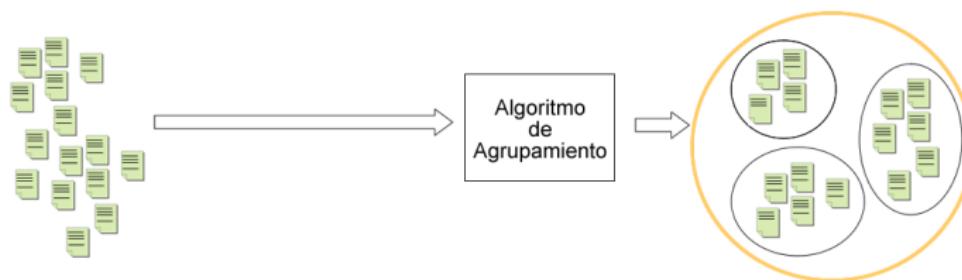
Ejemplo: Categorización Supervisada de Textos



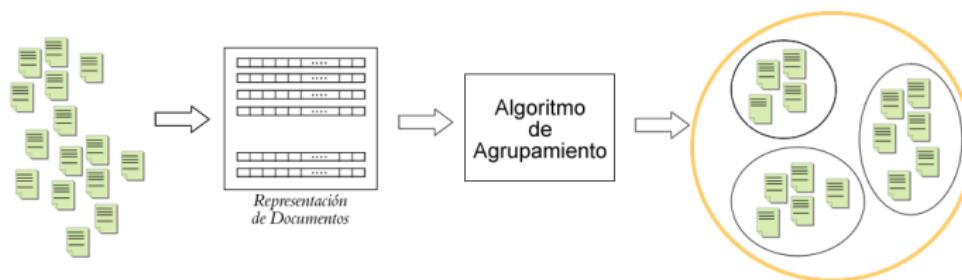
Ejemplo: Categorización Supervisada de Textos



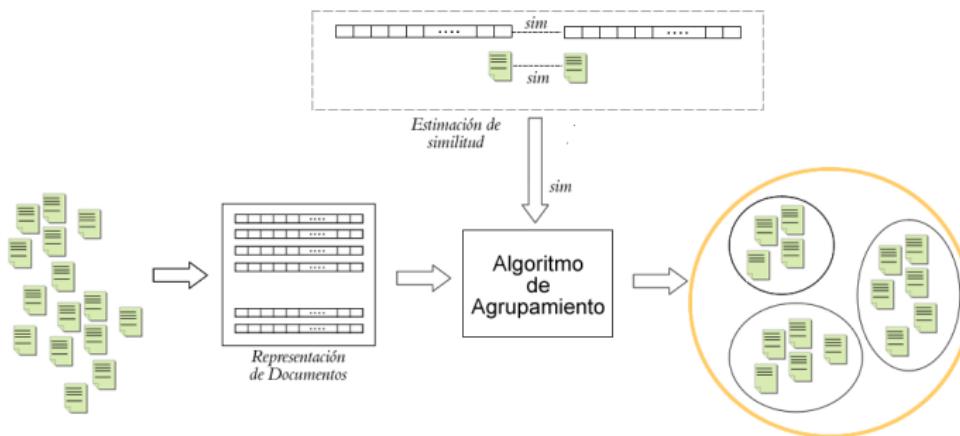
Agrupamiento de documentos



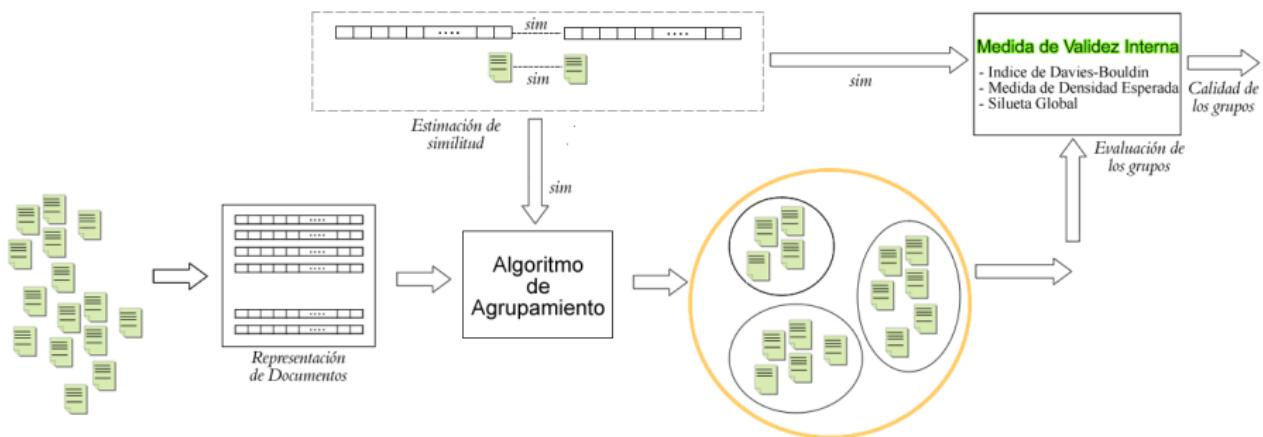
Agrupamiento de documentos



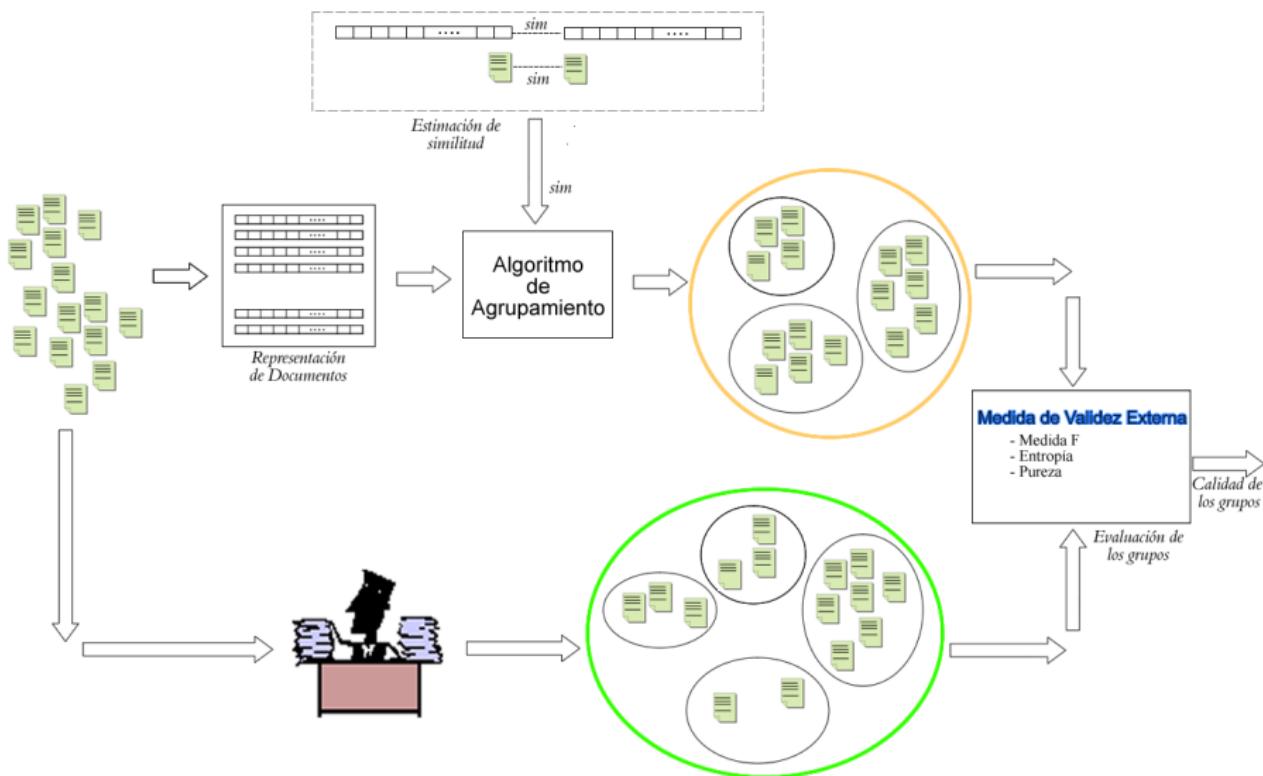
Agrupamiento de documentos



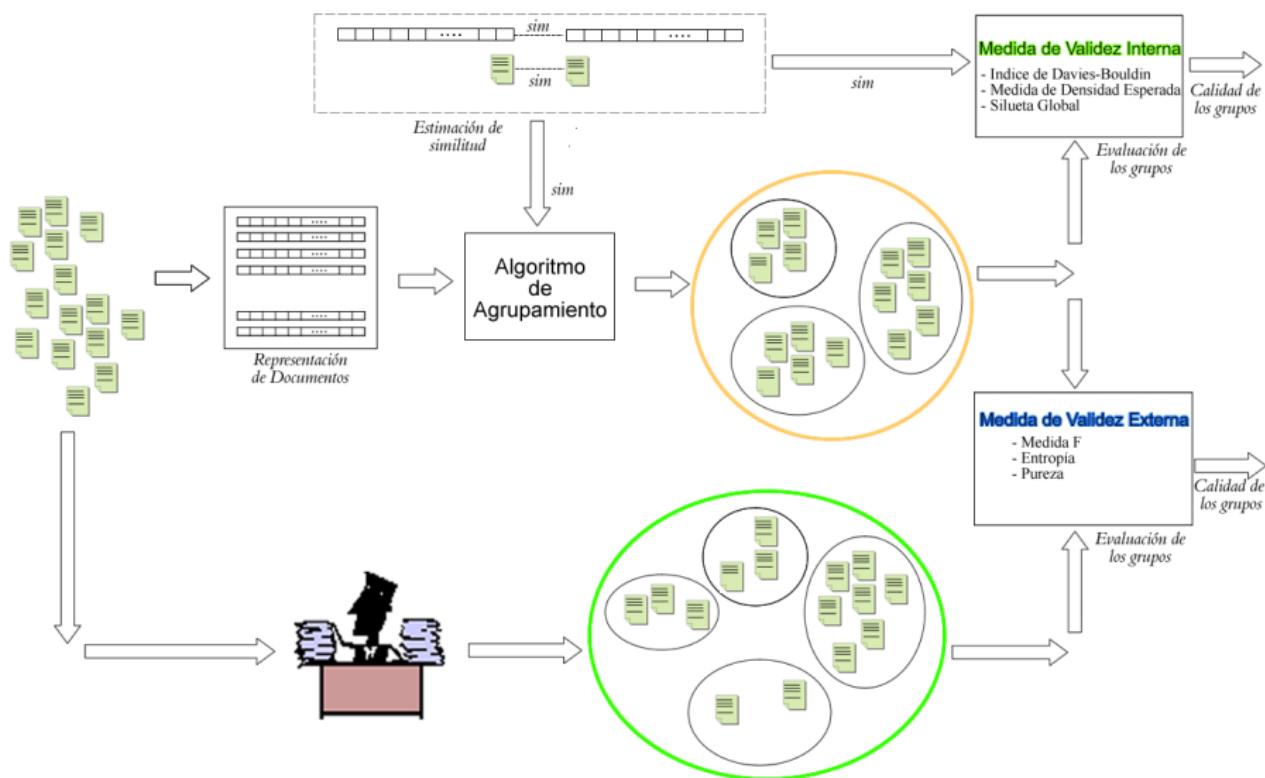
Agrupamiento de documentos



Agrupamiento de documentos



Agrupamiento de documentos



Modelización de tópicos



Extracción de Información (EI)

- Tarea que convierte texto **no (o semi) estructurado** en **representaciones estructuradas** (ej., **estructuras tabulares** para bases de datos).

Extracción de Información (EI)

- Tarea que convierte texto **no (o semi) estructurado** en **representaciones estructuradas** (ej., **estructuras tabulares** para bases de datos).
- Las estructuras obtenidas contienen **información útil y etiquetada** referida a **nombres/entidades, relaciones y eventos** identificados en el texto original.

Ejemplo de EI

EI de actividad terrorista a partir de un cable de noticia.

Dado el siguiente texto:

19 March – A bomb went off this morning near a power tower in San Salvador leaving a large part of the city without energy, but no casualties have been reported. According to unofficial sources, the bomb – allegedly detonated by urban guerrilla commandos – blew up a power tower in the northwestern part of San Salvador at 0650 (1250 GMT).

Ejemplo de EI (II)

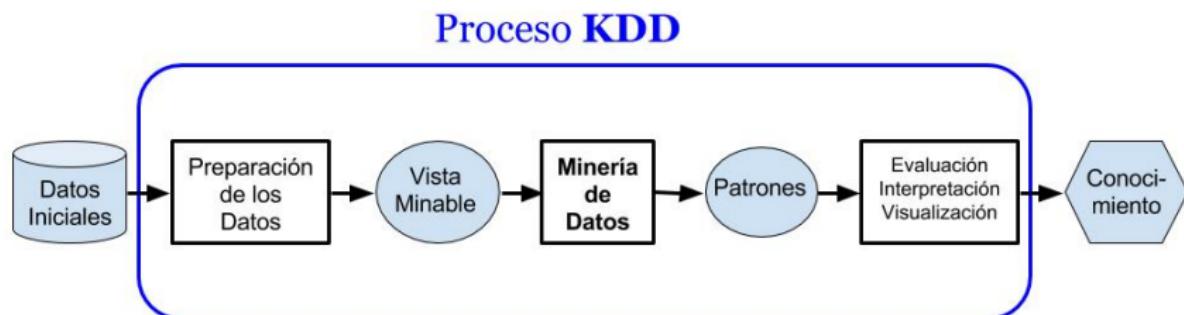
... se genera (“rellena”) el siguiente “template”:

Incident type	bombing
Date	March 19
Location	El Salvador: San Salvador (city)
Perpetrator	urban guerilla commandos
Physical target	power tower
Human target	-
Effect on physical target	destroyed
Effect on human target	no injury or death
Instrument	bomb

KDD a partir de textos

Sirve de guía para la organización de los contenidos de este curso.

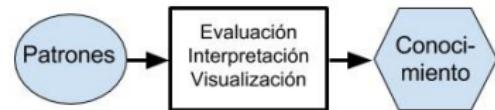
Fases del Proceso KDD



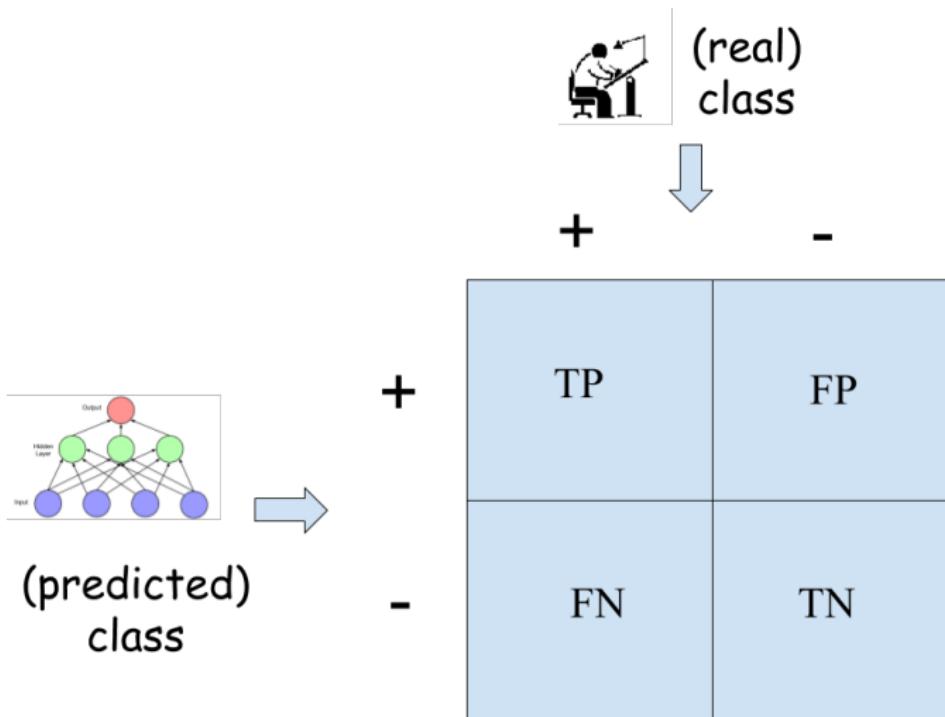
Fase de evaluación, interpretación y visualización

Sirve de guía para la organización de los contenidos de este curso.

Fases del Proceso KDD



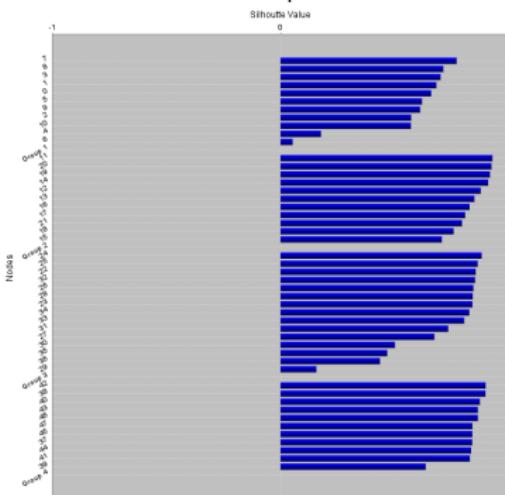
Evaluación (supervisada) clásica



Evaluando agrupamientos

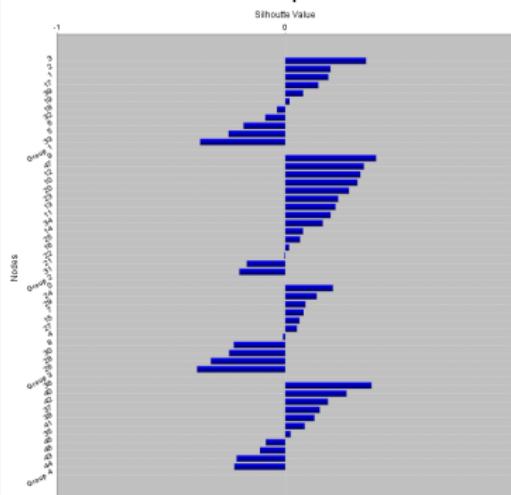
Agrupamiento bueno

Silhouette Graphic



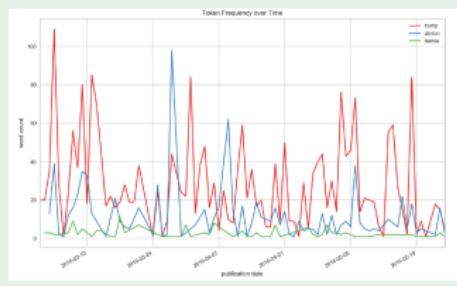
Agrupamiento malo

Silhouette Graphic



Visualizando información de textos

Frecuencia de palabras en el tiempo

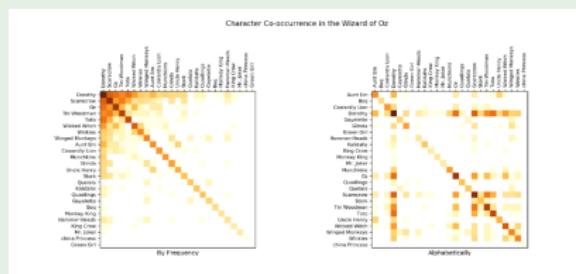


Grafo social de palabras

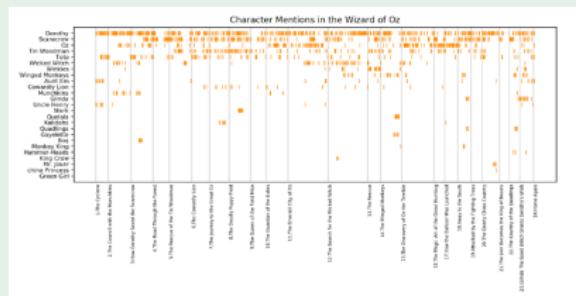


Visualizando información de textos

Matrices de co-ocurrencia

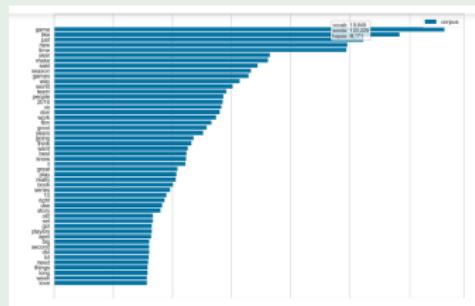


Gráficas de dispersión

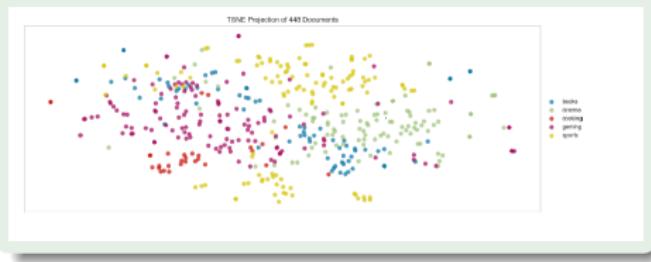


Visualizando información de textos

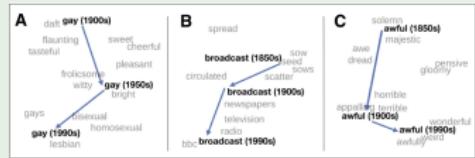
Frecuencias



Visualización con t-SNE



Evolución en el tiempo



Cercanía de embeddings



Algunas técnicas de pre-procesamiento

- ➊ Partición del texto
- ➋ Filtrado (“stop-words”, baja frecuencia)
- ➌ Normalización (mayúsculas, variaciones de uso)
- ➍ Truncado (“stemming”) y lematización (“lemmatization”)
- ➎ Etiquetado de las palabras
 - De Partes de la Oración (Part of Speech (**POS**) Tagging)
 - Desambiguación del Significado de las Palabras (**WSD**)
 - Reconocimiento de Entidades Nombradas (**NER**)

Partición del texto

Partición del texto “crudo”

Constituyen el pre-procesamiento **básico** en la mayoría de las aplicaciones de TM.

- ① **Partición:** proceso que convierte el texto crudo en componentes **significativas** (de acuerdo al caso):
 - ① capítulos
 - ② secciones
 - ③ párrafos
 - ④ sentencias
 - ⑤ palabras (e incluso sílabas)

Partición del texto

Partición del texto “crudo”

- ① **Partición:** proceso que convierte el texto crudo en componentes **significativas** (de acuerdo al caso):
 - ① capítulos
 - ② secciones
 - ③ párrafos
 - ④ sentencias
 - ⑤ palabras (e incluso sílabas)
- ② La forma más común y básica de partición es el proceso que convierte una **secuencia de caracteres** en una **secuencia de palabras** (o **tokens**) usualmente referida como **tokenización**.

Partición del texto

Tokenización

- ➊ Tokenización: tarea de **separar** (cortar/dividir) una **cadena de caracteres** en las **mínimas unidades lingüísticas identificables (tokens)**
- ➋ **Ejemplo**, dada la siguiente sentencia:

After sleeping for four hours, he decided to sleep for another four.

El **proceso de tokenización** produciría:

{“*After*” “*sleeping*” “*for*” “*four*” “*hours*” “*he*” “*decided*” “*to*” “*sleep*” “*for*” “*another*” “*four*”}.

Partición del texto

Tokens/Palabras

- 1 **Token:** secuencia de caracteres de un texto que es tratado como una unidad indivisible para su procesamiento.

Partición del texto

Tokens/Palabras

- ① **Token:** secuencia de caracteres de un texto que es tratado como una unidad indivisible para su procesamiento.
- ② **Regla simple** de tokenización:

Usar los “espacios en blanco” (caracteres de espacio, tabs y newlines) como separadores una vez que los símbolos de puntuación han sido removidos.

Partición del texto

Tokens/Palabras

- ① **Token:** secuencia de caracteres de un texto que es tratado como una unidad indivisible para su procesamiento.
- ② **Regla simple** de tokenización:

Usar los “espacios en blanco” (caracteres de espacio, tabs y newlines) como separadores una vez que los símbolos de puntuación han sido removidos.

- ③ El método estándar es utilizar **algoritmos determinísticos** basados en **expresiones regulares** que se compilan en autómatas de estado finito muy eficientes.

Partición del texto

Expresiones regulares

- Se pueden usar para tokenizar texto y tener mucho más control sobre el proceso.

Partición del texto

Expresiones regulares

- Se pueden usar para tokenizar texto y tener mucho más control sobre el proceso.
- Una **expresión regular** (ER) es una **notación algebraica** (un lenguaje) para caracterizar un **conjunto de cadenas**.

Partición del texto

Expresiones regulares

- Se pueden usar para tokenizar texto y tener mucho más control sobre el proceso.
- Una **expresión regular** (ER) es una **notación algebraica** (un lenguaje) para caracterizar un **conjunto de cadenas**.

Partición del texto

Meta-caracteres de expresiones regulares básicas

Operator	Behavior
.	Wildcard, matches any character
^abc	Matches some pattern <i>abc</i> at the start of a string
abc\$	Matches some pattern <i>abc</i> at the end of a string
[abc]	Matches one of a set of characters
[A-Z0-9]	Matches one of a range of characters
ed ing s	Matches one of the specified strings (disjunction)
*	Zero or more of previous item, e.g. a*, [a-z]* (also known as <i>Kleene Closure</i>)
+	One or more of previous item, e.g. a+, [a-z]+
?	Zero or one of the previous item (i.e. optional), e.g. a?, [a-z]?
{n}	Exactly <i>n</i> repeats where <i>n</i> is a non-negative integer
{n,}	At least <i>n</i> repeats
{,n}	No more than <i>n</i> repeats
{m,n}	At least <i>m</i> and no more than <i>n</i> repeats
a(b c)+	Parentheses that indicate the scope of the operators

Partición del texto

Algunos símbolos de expresiones regulares

Symbol	Function
\b	Word boundary (zero width)
\d	Any decimal digit (equivalent to [0-9])
\D	Any non-digit character (equivalent to [^0-9])
\s	Any whitespace character (equivalent to [\t\n\r\f\v])
\S	Any non-whitespace character (equivalent to [^\t\n\r\f\v])
\w	Any alphanumeric character (equivalent to [a-zA-Z0-9_])
\W	Any non-alphanumeric character (equivalent to [^a-zA-Z0-9_])
\t	The tab character
\n	The newline character

Partición del texto

Separando tokens usando espacios en blanco

In [1]:

```
import re
```

```
raw = """'When I'M a Duchess,' she said to herself, (not in a very  
hopeful tone though), 'I won't have any pepper in my kitchen AT  
ALL. Soup does very well without--Maybe it's always pepper that  
makes people hot-tempered,' """
```

```
print(raw.split()) #usando split como herramienta  
print(re.split(r' ', raw)) #con expresiones regulares (ojo)
```

Partición del texto

Separando tokens usando espacios en blanco

In [1]:

```
import re

raw = """'When I'M a Duchess,' she said to herself, (not in a very
hopeful tone though), 'I won't have any pepper in my kitchen AT
ALL. Soup does very well without--Maybe it's always pepper that
makes people hot-tempered,'"""

print(raw.split()) #usando split como herramienta
print(re.split(r' ', raw)) #con expresiones regulares (ojo)

["'When", "I'M", 'a', "Duchess,'", 'she', 'said', 'to', 'herself,', '
(not', 'in', 'a', 'very', 'hopeful', 'tone', 'though)', '', "'I",
"won't", 'have', 'any', 'pepper', 'in', 'my', 'kitchen', 'AT',
'ALL.', 'Soup', 'does', 'very', 'well', 'without--Maybe', "it's",
'always', 'pepper', 'that', 'makes', 'people', "hot-tempered,"]

["'When", "I'M", 'a', "Duchess,'", 'she', 'said', 'to', 'herself,', '
(not', 'in', 'a', 'very', '\n\nhopeful', 'tone', 'though)', '', "'I",
"won't", 'have', 'any', 'pepper', 'in', 'my', 'kitchen', 'AT',
'\n\nALL.', 'Soup', 'does', 'very', 'well', 'without--Maybe', "it's",
'always', 'pepper', 'that', '\nmakes', 'people', "hot-tempered,"]
```

Partición del texto

... mejorando la ER que separa tokens...

In [1]:

```
print(re.split(r'[ \t\n]+', raw))
```

Partición del texto

... mejorando la ER que separa tokens...

In [1]:

```
print(re.split(r'\t\n]+', raw))
```

```
['"When", "I'M", 'a', "Duchess,", 'she', 'said', 'to', 'herself,',  
'(not', 'in', 'a', 'very', 'hopeful', 'tone', 'though)', "'I",  
"won't", 'have', 'any', 'pepper', 'in', 'my', 'kitchen', 'AT',  
'ALL.', 'Soup', 'does', 'very', 'well', 'without--Maybe', "it's",  
'always', 'pepper', 'that', 'makes', 'people', "hot-tempered,"]
```

o usando la abreviatura \s (cualquier caracter de espacio en blanco)

In [1]:

```
print(re.split(r'\s+', raw))
```

Partición del texto

... mejorando la ER que separa tokens...

In [1]:

```
print(re.split(r'\t\n]+', raw))
```

```
["'When", "I'M", 'a', "Duchess,'", 'she', 'said', 'to', 'herself,',  
'(not', 'in', 'a', 'very', 'hopeful', 'tone', 'though)', 'I",  
"won't", 'have', 'any', 'pepper', 'in', 'my', 'kitchen', 'AT',  
'ALL.', 'Soup', 'does', 'very', 'well', 'without--Maybe', "it's",  
'always', 'pepper', 'that', 'makes', 'people', "hot-tempered,'"]
```

o usando la abreviatura \s (cualquier caracter de espacio en blanco)

In [1]:

```
print(re.split(r'\s+', raw))
```

```
["'When", "I'M", 'a', "Duchess,'", 'she', 'said', 'to', 'herself,',  
'(not', 'in', 'a', 'very', 'hopeful', 'tone', 'though)', 'I",  
"won't", 'have', 'any', 'pepper', 'in', 'my', 'kitchen', 'AT',  
'ALL.', 'Soup', 'does', 'very', 'well', 'without--Maybe', "it's",  
'always', 'pepper', 'that', 'makes', 'people', "hot-tempered,'"]
```

Partición del texto

... otras variantes de separación ...

Cosas que no sean **caracteres de palabras** (CPs):

In [1]:

```
print(re.split(r'\W+', raw))
```

Partición del texto

... otras variantes de separación ...

Cosas que no sean **caracteres de palabras** (CPs):

In [1]:

```
print(re.split(r'\W+', raw))
```

```
['', 'When', 'I', 'M', 'a', 'Duchess', 'she', 'said', 'to', 'herself',
'not', 'in', 'a', 'very', 'hopeful', 'tone', 'though', 'I', 'won',
't', 'have', 'any', 'pepper', 'in', 'my', 'kitchen', 'AT', 'ALL',
'Soup', 'does', 'very', 'well', 'without', 'Maybe', 'it', 's', 'always',
'pepper', 'that', 'makes', 'people', 'hot', 'tempered', '']
```

o buscando las palabras en lugar de los separadores:

In [1]:

```
print(re.findall(r'\w+', raw))
```

Partición del texto

... otras variantes de separación ...

Cosas que no sean **caracteres de palabras** (CPs):

In [1]:

```
print(re.split(r'\W+', raw))
```

```
['', 'When', 'I', 'M', 'a', 'Duchess', 'she', 'said', 'to', 'herself',
'not', 'in', 'a', 'very', 'hopeful', 'tone', 'though', 'I', 'won',
't', 'have', 'any', 'pepper', 'in', 'my', 'kitchen', 'AT', 'ALL',
'Soup', 'does', 'very', 'well', 'without', 'Maybe', 'it', 's', 'always',
'pepper', 'that', 'makes', 'people', 'hot', 'tempered', '']
```

o buscando las palabras en lugar de los separadores:

In [1]:

```
print(re.findall(r'\w+', raw))
```

```
['When', 'I', 'M', 'a', 'Duchess', 'she', 'said', 'to', 'herself',
'not', 'in', 'a', 'very', 'hopeful', 'tone', 'though', 'I', 'won',
't', 'have', 'any', 'pepper', 'in', 'my', 'kitchen', 'AT', 'ALL',
'Soup', 'does', 'very', 'well', 'without', 'Maybe', 'it', 's',
'always', 'pepper', 'that', 'makes', 'people', 'hot', 'tempered']
```

Partición del texto

... y buscando patrones más complejos ...

CPs o algo que no es un espacio en blanco seguido de CPs:

In [1]:

```
print(re.findall(r'\w+|\S\w*', raw))
```

Partición del texto

... y buscando patrones más complejos ...

CPs o algo que no es un espacio en blanco seguido de CPs:

In [1]:

```
print(re.findall(r'\w+|\S\w*', raw))
```

```
["'When", "'I", "'M", 'a', 'Duchess', ',', "'", 'she', 'said', 'to',  
'herself', ',', '(not', 'in', 'a', 'very', 'hopeful', 'tone', 'though'  
)', ',', "'I", 'won', "'t", 'have', 'any', 'pepper', 'in', 'my',  
'kitchen', 'AT', 'ALL', '.', 'Soup', 'does', 'very', 'well', 'without'  
'-', '-Maybe', 'it', "'s", 'always', 'pepper', 'that', 'makes', 'peopl  
'hot', '-tempered', ',', ""]
```

Filtrado de palabras

Filtrado

Se **remueven palabras** (tokens) de determinado tipo o que no cumplen ciertas condiciones:

- **Palabras de paro** (“stop-words”): palabras con escasa (o nula) información de contenido (artículos, conjunciones, preposiciones, etc).
 - En ciertas tareas no son eliminadas (ej: se requiere capturar **estilo**)
 - Existen diccionarios **específicos del lenguaje**
 - Se puede “aproximar” este efecto con eliminación de palabras **demasiado frecuentes** o el uso de **normalizaciones de pesos** que las **penalizan (idf)**
- **Umbrales** de frecuencia mínimo (ver métodos de filtrado en teoría de **reducción de dimensionalidad**).
- Símbolos o elementos **especiales** (símbolos de puntuación, números, etc)

Normalización de palabras

Normalizaciones (mayúsculas y variaciones de uso)

En este contexto, **normalizar** es llevar variaciones de tokens a una forma común.

- El uso de la **mayúscula** puede definir el significado de un término. **Bob** puede ser un **verbo** usado al comienzo de una sentencia o el nombre de una persona.

Normalización de palabras

Normalizaciones (mayúsculas y variaciones de uso)

En este contexto, **normalizar** es llevar variaciones de tokens a una forma común.

- El uso de la **mayúscula** puede definir el significado de un término. **Bob** puede ser un **verbo** usado al comienzo de una sentencia o el nombre de una persona.
- Regla usual: palabras en el comienzo de sentencias, títulos y encabezados de sección se **convierten a minúscula**. El resto se mantiene como está.
- Otra casos que suelen normalizarse: pequeñas variaciones del mismo token que refieren a la misma palabra/concepto: **colour/color**, **naive/naïve**, **US/USA**.
- Solución: tablas con posibles variaciones y su forma estandarizada.

Normalización de palabras

Lematización y truncado

- Se pueden ver como otra forma de **normalización / standarización**
- Buscan reducir las **formas infleccionales y derivadas** de las palabras, obteniendo las **formas bases** comunes.

Métodos de *lematización*

- Mapean palabras a su forma **base** o de **diccionario** (lema).
- En formas verbales busca el **infinitivo** y en sustantivos su forma **singular**.
- Requieren del uso de rotuladores (p. ej. POS tagger)
- Ejemplo: **saw** ⇒ **see** o **saw** (dependiendo de si es reconocido como sustantivo o verbo)

Normalización de palabras

Lematización y truncado

- Se pueden ver como otra forma de **normalización / standarización**
- Buscan reducir las **formas infleccionales y derivadas** de las palabras, obteniendo las **formas bases** comunes.

Métodos de *truncado* (“stemming”)

- Considerados por algunos como una **lematización básica**.
- Más sencillo y económico que la obtención de lemas.
- Obtiene formas básicas de las palabras a partir de la *poda* de sufijos (“s” en sustantivos, “ing” en verbos, etc.)
- El “stem” o raíz de la palabra intenta representar palabras con igual (o similar significado).

Normalización de palabras

Ejemplos de truncado

El algoritmo de stemming (para inglés) más conocido es el alg. de **Porter**. Sitio oficial:

<http://tartarus.org/martin/PorterStemmer/>

Ejemplos con Porter

- Ejemplo 1: Porter convierte: **operate operating operates operation operative operatives operational** en **oper**.
- Ejemplo 2: Ojo, Porter podría convertir **saw** ⇒ **s**.
- Buena comparación de stemming y lemmatization:
<http://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html>

Normalización de palabras

Ejemplo completo con Porter

This was not the map we found in Billy
Bones's chest, but an accurate copy, complete
in all things-names and heights and
soundings-with the single exception of the red
crosses and the written notes.

produce la siguiente salida con palabras truncadas

Thi wa not the map we found in Billi Bone s
chest but an accur copi complet in all thing
name and height and sound with the singl
except of the red cross and the written note

Normalización de palabras

Comparación truncado (nltk) vs lematización(spacy)

In [1]:

```
import spacy
import nltk

# cargar el modelo del lenguaje inglés de spacy
en_nlp = spacy.load('en')
doc = u"I saw there some saws to cut the tree"
# instanciar el "stemmer" de Porter de nltk
stemmer = nltk.stem.PorterStemmer()
# tokenizar documento con spacy
doc_spacy = en_nlp(doc)
# imprimir lemas encontrados por spacy
print("Lematización:")
print([token.lemma_ for token in doc_spacy])
# imprimir tokens obtenidos con el stemmer de Porter
print("Truncado:")
print([stemmer.stem(token.norm_.lower()) for token in doc_spacy])
```

Normalización de palabras

Comparación truncado (nltk) vs lematización(spacy)

In [1]:

```
import spacy
import nltk

# cargar el modelo del lenguaje inglés de spacy
en_nlp = spacy.load('en')
doc = u"I saw there some saws to cut the tree"
# instanciar el "stemmer" de Porter de nltk
stemmer = nltk.stem.PorterStemmer()
# tokenizar documento con spacy
doc_spacy = en_nlp(doc)
# imprimir lemas encontrados por spacy
print("Lematización:")
print([token.lemma_ for token in doc_spacy])
# imprimir tokens obtenidos con el stemmer de Porter
print("Truncado:")
print([stemmer.stem(token.norm_.lower()) for token in doc_spacy])

Lematización:
['-PRON-', 'see', 'there', 'some', 'saw', 'to', 'cut', 'the', 'tree']
Truncado:
['i', 'saw', 'there', 'some', 'saw', 'to', 'cut', 'the', 'tree']
```

Etiquetado de palabras

Etiquetado de las Categorías Gramaticales (ECG)

Las palabras de una oración pueden ser etiquetadas/categorizadas de acuerdo al **rol** que tienen en el contexto de una sentencia: **sustantivo, verbo, adjetivo, adverbio**, etc.

Etiquetado de palabras

Etiquetado de las Categorías Gramaticales (ECG)

Las palabras de una oración pueden ser etiquetadas/categorizadas de acuerdo al **rol** que tienen en el contexto de una sentencia: **sustantivo, verbo, adjetivo, adverbio**, etc.

Ejemplo, en:

*Si usted **tapa** la olla azul, necesitará una **tapa** grande.*

El primer uso de **tapa** corresponde a un **verbo**, y el segundo a un **sustantivo**.

Etiquetado de palabras

Etiquetado de las Categorías Gramaticales (ECG)

Las palabras de una oración pueden ser etiquetadas/categorizadas de acuerdo al **rol** que tienen en el contexto de una sentencia: **sustantivo, verbo, adjetivo, adverbio**, etc.

Ejemplo, en:

*Si usted **tapa** la olla azul, necesitará una **tapa** grande.*

El primer uso de **tapa** corresponde a un **verbo**, y el segundo a un **sustantivo**.

Estas categorías de una palabra de acuerdo al contexto de uso se las conoce como **categorías gramaticales, clases de palabras o partes de la oración** (en inglés **Part-of-Speech (POS)**)

Etiquetado de palabras

Etiquetado de las Categorías Gramaticales (ECG) (II)

Definición

El **Etiquetado de las Categorías Gramaticales (ECG)** (en inglés **Part-of-Speech (POS) Tagging**) es la tarea de asignar a las palabras distintas categorías gramaticales de acuerdo al rol que tienen en las sentencias en que aparecen.

Etiquetado de palabras

Etiquetado de las Categorías Gramaticales (ECG) (II)

Definición

El **Etiquetado de las Categorías Gramaticales (ECG)** (en inglés **Part-of-Speech (POS) Tagging**) es la tarea de asignar a las palabras distintas categorías gramaticales de acuerdo al rol que tienen en las sentencias en que aparecen.

Ejemplo: las palabras de la sentencia

The grand jury commented on a number of other topics .

Etiquetado de palabras

Etiquetado de las Categorías Gramaticales (ECG) (II)

Definición

El **Etiquetado de las Categorías Gramaticales (ECG)** (en inglés **Part-of-Speech (POS) Tagging**) es la tarea de asignar a las palabras distintas categorías gramaticales de acuerdo al rol que tienen en las sentencias en que aparecen.

Ejemplo: las palabras de la sentencia

The grand jury commented on a number of other topics .

podrían ser etiquetadas con las siguientes categorías por un sistema de ECG:

The/DT** grand/**JJ** jury/**NN** commented/**VBD** on/**IN** a/**DT** number/**NN** of/**IN** other/**JJ** topics/**NNS** ./.**

Etiquetado de palabras

Etiquetado de las Categorías Gramaticales (ECG) (III)

The/**DT** grand/**JJ** jury/**NN** commented/**VBD** on/**IN**
a/**DT** number/**NN** of/**IN** other/**JJ** topics/**NNS** ./.

- **DT** (determinador), **JJ** (adjetivo), **NN** (sustantivo), **IN** (preposición), **VBD** (verbo en pasado) y **NNS** (sustantivo plural) son etiquetas usuales en una sentencia.
- Sin embargo, considerando distintas categorías y subcategorías, se han propuesto distintos **tagsets**:
 - 1 Penn Treebank (45 etiquetas)
 - 2 Brown corpus (87 etiquetas)
 - 3 C7 tagset (146 etiquetas)

Etiquetado de palabras

Etiquetas para el inglés (Penn Treebank)

Tag	Description	Example	Tag	Description	Example
CC	Coordin. Conjunction	<i>and, but, or</i>	SYM	Symbol	+,%,&
CD	Cardinal number	<i>one, two, three</i>	TO	"to"	<i>to</i>
DT	Determiner	<i>a, the</i>	UH	Interjection	<i>ah, oops</i>
EX	Existential 'there'	<i>there</i>	VB	Verb, base form	<i>eat</i>
FW	Foreign word	<i>mea culpa</i>	VBD	Verb, past tense	<i>ate</i>
IN	Preposition/sub-conj	<i>of, in, by</i>	VBG	Verb, gerund	<i>eating</i>
JJ	Adjective	<i>yellow</i>	VBN	Verb, past participle	<i>eaten</i>
JJR	Adj., comparative	<i>bigger</i>	VBP	Verb, non-3sg pres	<i>eat</i>
JJS	Adj., superlative	<i>wildest</i>	VBZ	Verb, 3sg pres	<i>eats</i>
LS	List item marker	<i>1, 2, One</i>	WDT	Wh-determiner	<i>which, that</i>
MD	Modal	<i>can, should</i>	WP	Wh-pronoun	<i>what, who</i>
NN	Noun, sing. or mass	<i>llama</i>	WP\$	Possessive wh-	<i>whose</i>
NNS	Noun, plural	<i>llamas</i>	WRB	Wh-adverb	<i>how, where</i>
NNP	Proper noun, singular	<i>IBM</i>	\$	Dollar sign	\$
NNPS	Proper noun, plural	<i>Carolinas</i>	#	Pound sign	#
PDT	Predeterminer	<i>all, both</i>	"	Left quote	' or "
POS	Possessive ending	<i>'s</i>	"	Right quote	' or "
PRP	Personal pronoun	<i>I, you, he</i>	(Left parenthesis	[, (, {, <
PRP\$	Possessive pronoun	<i>your, one's</i>)	Right parenthesis],), }, >
RB	Adverb	<i>quickly, never</i>	,	Comma	,
RBR	Adverb, comparative	<i>faster</i>	.	Sentence-final punc	. ! ?
RBS	Adverb, superlative	<i>fastest</i>	:	Mid-sentence punc	: ; ... - -
RP	Particle	<i>up, off</i>			

Etiquetado de palabras

Ejemplo 1: POS tagging (Python)

In [1]:

```
from nltk import pos_tag, wordpunct_tokenize
text = "The old building was demolished. Tomorrow, they will begin
building a new one"
pos_tag(wordpunct_tokenize(text))
```

Etiquetado de palabras

Ejemplo 1: POS tagging (Python)

In [1]:

```
from nltk import pos_tag, wordpunct_tokenize
text = "The old building was demolished. Tomorrow, they will begin
building a new one"
pos_tag(wordpunct_tokenize(text))
```

Out [1]:

```
[('The', 'DT'),
 ('old', 'JJ'),
 ('building', 'NN'),
 ('was', 'VBD'),
 ('demolished', 'VBN'),
 ('.', '.'),
 ('Tomorrow', 'NNP'),
 ('', ''),
 ('they', 'PRP'),
 ('will', 'MD'),
 ('begin', 'VB'),
 ('building', 'VBG'),
 ('a', 'DT'),
 ('new', 'JJ'),
 ('one', 'CD')]
```

Etiquetado de palabras

Ejemplo 2: POS tagging (Python)

In [1]:

```
text = "The grand jury commented on a number of other topics."  
pos_tag(wordpunct_tokenize(text))
```

Etiquetado de palabras

Ejemplo 2: POS tagging (Python)

In [1]:

```
text = "The grand jury commented on a number of other topics."  
pos_tag(wordpunct_tokenize(text))
```

Out [1]:

```
[('The', 'DT'),  
 ('grand', 'JJ'),  
 ('jury', 'NN'),  
 ('commented', 'VBD'),  
 ('on', 'IN'),  
 ('a', 'DT'),  
 ('number', 'NN'),  
 ('of', 'IN'),  
 ('other', 'JJ'),  
 ('topics', 'NNS'),  
 ('.', '.')]
```

Desambiguación del Significado de las Palabras

Definición

En inglés **Word Sense Disambiguation (WSD)** trata de resolver la ambigüedad en el significado de las palabras en base al contexto en que éstas aparecen.

Etiquetado de palabras

Desambiguación del Significado de las Palabras

Definición

En inglés **Word Sense Disambiguation (WSD)** trata de resolver la ambigüedad en el significado de las palabras en base al contexto en que éstas aparecen.

Ejemplo, la palabra “banco” ...

Frase		Significado
Perdí la mañana en el banco pagando impuestos.	⇒	Institución Financiera
Desde el barco vi el banco de peces.	⇒	Cardumen
Sentado en un banco suspiraba.	⇒	Mueble
Las donaciones se están recibiendo en el banco de sangre.	⇒	Establecimiento Médico

Etiquetado de palabras

Word Sense Disambiguation (WSD)

Desambigüación (Word Sense Disambiguation (WSD))

Procedimiento utilizado para decidir los significados de las palabras a partir del contexto que las rodea.

KDD a partir de textos



Etiquetado de palabras

Pre-procesamiento



La ontología Wordnet

Etiquetado de palabras

La ontología Wordnet

- Concebido como un diccionario electrónico.

Etiquetado de palabras

La ontología Wordnet

- Concebido como un diccionario electrónico.
- El contenido se organiza mediante una base de datos léxica, donde se agrupan conjuntos de palabras (nombres, verbos, adjetivos y adverbios) en grupos de sinónimos llamados **synsets**.

Etiquetado de palabras

La ontología Wordnet

- Concebido como un diccionario electrónico.
- El contenido se organiza mediante una base de datos léxica, donde se agrupan conjuntos de palabras (nombres, verbos, adjetivos y adverbios) en grupos de sinónimos llamados **synsets**.
- Por lo tanto, un **synset** es un conjunto de todas las palabras que expresan un mismo concepto.

Etiquetado de palabras

La ontología Wordnet

- Concebido como un diccionario electrónico.
- El contenido se organiza mediante una base de datos léxica, donde se agrupan conjuntos de palabras (nombres, verbos, adjetivos y adverbios) en grupos de sinónimos llamados **synsets**.
- Por lo tanto, un **synset** es un conjunto de todas las palabras que expresan un mismo concepto.
- Dentro de WN, cada synset se codifica con un **número único** que representa un **concepto distinto**.

Etiquetado de palabras

La ontología Wordnet

- Concebido como un diccionario electrónico.
- El contenido se organiza mediante una base de datos léxica, donde se agrupan conjuntos de palabras (nombres, verbos, adjetivos y adverbios) en grupos de sinónimos llamados **synsets**.
- Por lo tanto, un **synset** es un conjunto de todas las palabras que expresan un mismo concepto.
- Dentro de WN, cada synset se codifica con un **número único** que representa un **concepto distinto**.
- Entre los **synsets** existen conexiones que expresan **relaciones** semánticas, conceptuales o léxicas.

Etiquetado de palabras

Relaciones definidas en Wordnet

- **Sinonimia:** car ⇒ automobile

Etiquetado de palabras

Relaciones definidas en Wordnet

- Sinonimia: car ⇒ automobile
- Antonimia: clean ⇒ dirty

Etiquetado de palabras

Relaciones definidas en Wordnet

- **Sinonimia:** car ⇒ automobile
- **Antonimia:** clean ⇒ dirty
- **Hiponimia** (TIPOS DE): cat ⇒ domestic cat, cat ⇒ wildcat

Etiquetado de palabras

Relaciones definidas en Wordnet

- **Sinonimia:** car ⇒ automobile
- **Antonimia:** clean ⇒ dirty
- **Hipónimia** (TIPOS DE): cat ⇒ domestic cat, cat ⇒ wildcat
- **Hiperónimia** (ES UN TIPO DE): pine ⇒ tree, cat ⇒ feline
⇒ carnivore

Etiquetado de palabras

Relaciones definidas en Wordnet

- **Sinonimia:** car ⇒ automobile
- **Antonimia:** clean ⇒ dirty
- **Hipónimia** (TIPOS DE): cat ⇒ domestic cat, cat ⇒ wildcat
- **Hiperónimia** (ES UN TIPO DE): pine ⇒ tree, cat ⇒ feline
⇒ carnivore
- **Meronimia** (PARTES DE): foot ⇒ toe.

Etiquetado de palabras

Relaciones definidas en Wordnet

- **Sinonimia:** car ⇒ automobile
- **Antonimia:** clean ⇒ dirty
- **Hipónimia** (TIPOS DE): cat ⇒ domestic cat, cat ⇒ wildcat
- **Hiperónimia** (ES UN TIPO DE): pine ⇒ tree, cat ⇒ feline
⇒ carnivore
- **Meronimia** (PARTES DE): foot ⇒ toe.
- **Holonimia:** (ES UNA PARTE DE): toe ⇒ foot, eye ⇒ face

Etiquetado de palabras

Relaciones definidas en Wordnet

- **Sinonimia:** car ⇒ automobile
- **Antonimia:** clean ⇒ dirty
- **Hiponimia** (TIPOS DE): cat ⇒ domestic cat, cat ⇒ wildcat
- **Hiperonimia** (ES UN TIPO DE): pine ⇒ tree, cat ⇒ feline
⇒ carnivore
- **Meronimia** (PARTES DE): foot ⇒ toe.
- **Holonimia:** (ES UNA PARTE DE): toe ⇒ foot, eye ⇒ face
- **Troponimia:** es **hiponimia** a nivel verbos. swim ⇒ crawl

Etiquetado de palabras

Relaciones definidas en Wordnet

- **Sinonimia:** car ⇒ automobile
- **Antonimia:** clean ⇒ dirty
- **Hiponimia** (TIPOS DE): cat ⇒ domestic cat, cat ⇒ wildcat
- **Hiperonimia** (ES UN TIPO DE): pine ⇒ tree, cat ⇒ feline
⇒ carnivore
- **Meronimia** (PARTES DE): foot ⇒ toe.
- **Holonimia:** (ES UNA PARTE DE): toe ⇒ foot, eye ⇒ face
- **Troponimia:** es **hiponimia** a nivel verbos. swim ⇒ crawl
- **Entailment:** es la inferencia lógica. snore ⊨ sleep

Etiquetado de palabras

WSD y Wordnet

- La tarea de **WSD** es básicamente una de **clasiﬁcación**:

Etiquetado de palabras

WSD y Wordnet

- La tarea de **WSD** es básicamente una de **clasificación**:
*Dada una palabra, asignar un **sentido único** a la misma, de acuerdo a su contexto (palabras que la rodean) en la oración.*
- Este **sentido único** está dado por el identificador (único) de un **synset** de la ontología Wordnet.
- Un **synset** (por **conjunto de sinónimos**) es un conjunto de sentidos/significados de palabras que representan cosas que son **sinónimos** (o casi).
- Así, un **synset** representa un **concepto**, aquel representado como la lista de los significados de palabras que se usan para expresar dicho concepto.

Etiquetado de palabras

Ejemplo: sentidos en WN de la palabra “bass”

The noun “bass” has 8 senses in WordNet.

1. bass¹ - (the lowest part of the musical range)
2. bass², bass part¹ - (the lowest part in polyphonic music)
3. bass³, basso¹ - (an adult male singer with the lowest voice)
4. sea bass¹, bass⁴ - (the lean flesh of a saltwater fish of the family Serranidae)
5. freshwater bass¹, bass⁵ - (any of various North American freshwater fish with
lean flesh (especially of the genus Micropterus))
6. bass⁶, bass voice¹, basso² - (the lowest adult male singing voice)
7. bass⁷ - (the member with the lowest range of a family of musical instruments)
8. bass⁸ - (nontechnical name for any of numerous edible marine and
freshwater spiny-finned fishes)

The adjective “bass” has 1 sense in WordNet.

1. bass¹, deep⁶ - (having or denoting a low vocal or instrumental range)
“a deep voice”; “a bass voice is lower than a baritone voice”;
“a bass clarinet”

Etiquetado de palabras

Sentidos y synsets

- *bass* tiene 8 sentidos como sustantivo y 1 como adjetivo
- Aquí se pueden identificar synsets como $\{bass^1, deep^6\}$ y $\{bass^6, bassvoice^1, basso^2\}$
- Con **NLTK** se pueden obtener todos los sentidos de una palabra en Wordnet:

Etiquetado de palabras

Sentidos y synsets

- *bass* tiene 8 sentidos como sustantivo y 1 como adjetivo
- Aquí se pueden identificar synsets como $\{bass^1, deep^6\}$ y $\{bass^6, bassvoice^1, basso^2\}$
- Con NLTK se pueden obtener todos los sentidos de una palabra en Wordnet:

In [1]:

```
from nltk.corpus import wordnet as wn  
wn.synsets('bass')
```

Etiquetado de palabras

Sentidos y synsets

- *bass* tiene 8 sentidos como sustantivo y 1 como adjetivo
- Aquí se pueden identificar synsets como $\{bass^1, deep^6\}$ y $\{bass^6, bassvoice^1, basso^2\}$
- Con **NLTK** se pueden obtener todos los sentidos de una palabra en Wordnet:

In [1]:

```
from nltk.corpus import wordnet as wn
wn.synsets('bass')
```

Out[1]:

```
[Synset('bass.n.01'),
 Synset('bass.n.02'),
 Synset('bass.n.03'),
 Synset('sea_bass.n.01'),
 Synset('freshwater_bass.n.01'),
 Synset('bass.n.06'),
 Synset('bass.n.07'),
 Synset('bass.n.08'),
 Synset('bass.s.01')]
```

Etiquetado de palabras

WSD (NLTK)

- Desambiguar el sentido de la palabra (WSD) no es más que dar el **lema** de la palabra, su **rol/categoría gramatical** y **el contexto** en que está usada.
- El resultado es el **synset/concepto** (únívoco) que corresponde a este uso del lema:
- **Ejemplo:** usando para WSD el algoritmo de **Lesk**

Etiquetado de palabras

WSD (NLTK)

- Desambiguar el sentido de la palabra (WSD) no es más que dar el **lema** de la palabra, su **rol/categoría gramatical** y **el contexto** en que está usada.
- El resultado es el **synset/concepto** (únívoco) que corresponde a este uso del lema:
- **Ejemplo:** usando para WSD el algoritmo de **Lesk**

In [1]:

```
lesk(['I', 'went', 'to', 'the', 'bank', 'to', 'deposit', 'money', '.']
      'bank', 'n'))
```

Etiquetado de palabras

WSD (NLTK)

- Desambiguar el sentido de la palabra (WSD) no es más que dar el **lema** de la palabra, su **rol/categoría gramatical** y **el contexto** en que está usada.
- El resultado es el **synset/concepto** (únívoco) que corresponde a este uso del lema:
- **Ejemplo:** usando para WSD el algoritmo de **Lesk**

In [1]:

```
lesk(['I', 'went', 'to', 'the', 'bank', 'to', 'deposit', 'money', '.']
      'bank', 'n')
```

Out[1]:

```
Synset('savings_bank.n.02')
```

Reconocimiento de Entidades Nombradas (REN)

Características

- Subtarea de Extracción de Información.

Reconocimiento de Entidades Nombradas (REN)

Características

- Subtarea de Extracción de Información.
- Consiste en la **ubicación** y **clasificación** de secuencias de palabras dentro de un texto, en categorías tales como **nombres de personas, lugares, organizaciones, cantidades**, etc.

Reconocimiento de Entidades Nombradas (REN)

Características

- Subtarea de Extracción de Información.
- Consiste en la **ubicación** y **clasificación** de secuencias de palabras dentro de un texto, en categorías tales como **nombres de personas, lugares, organizaciones, cantidades**, etc.
- El resultado de este proceso es un documento con etiquetas que identifican el comienzo y fin de las entidades nombradas.

Reconocimiento de Entidades Nombradas (REN)

Características

- Subtarea de Extracción de Información.
- Consiste en la **ubicación** y **clasificación** de secuencias de palabras dentro de un texto, en categorías tales como **nombres de personas, lugares, organizaciones, cantidades**, etc.
- El resultado de este proceso es un documento con etiquetas que identifican el comienzo y fin de las entidades nombradas.
- Ejemplo: “Jim bought 300 shares of Acme Corp. in 2006”
⇒ <ENAMEX TYPE=“PERSON”>Jim</ENAMEX> bought <NUMEX TYPE=“QUANTITY”>300</NUMEX> shares of <ENAMEX TYPE=“ORGANIZATION”>Acme Corp.</ENAMEX> in <TIMEX TYPE=“DATE”>2006</TIMEX>.

Etiquetado de palabras

Entidades Nombradas (EN)

Entidades nombradas

- Secuencia de palabras que refiere a una entidad específica del mundo real mediante un nombre propio: una persona, una ubicación, una organización.

Etiquetado de palabras

Entidades Nombradas (EN)

Entidades nombradas

- Secuencia de palabras que refiere a una entidad específica del mundo real mediante un nombre propio: una persona, una ubicación, una organización.
- También incluye otras cosas que no son estrictamente entidades nombradas (fechas), valores monetarios, etc

Etiquetado de palabras

Entidades Nombradas (EN)

Entidades nombradas

- Secuencia de palabras que refiere a una entidad específica del mundo real mediante un nombre propio: una persona, una ubicación, una organización.
- También incluye otras cosas que no son estrictamente entidades nombradas (fechas), valores monetarios, etc

Tipos y ejemplos de EN genéricas

Type	Tag	Sample Categories	Example sentences
People	PER	people, characters	Turing is a giant of computer science.
Organization	ORG	companies, sports teams	The IPCC warned about the cyclone.
Location	LOC	regions, mountains, seas	The Mt. Sanitas loop is in Sunshine Canyon.
Geo-Political	GPE	countries, states, provinces	Palo Alto is raising the fees for parking.
Entity			
Facility	FAC	bridges, buildings, airports	Consider the Tappan Zee Bridge.
Vehicles	VEH	planes, trains, automobiles	It was a classic Ford Falcon.

Etiquetado de palabras

REN en NLTK-Python

Se puede usar la función `nltk.ne_chunk()`, que toma como entrada una sentencia con los POS tags.

Usaremos una sentencia del corpus TreeBank

In [1]:

```
sent = nltk.corpus.treebank.tagged_sents()[22]
sent
```

Etiquetado de palabras

REN en NLTK-Python

Se puede usar la función `nltk.ne_chunk()`, que toma como entrada una sentencia con los POS tags.

Usaremos una sentencia del corpus TreeBank

In [1]:

```
sent = nltk.corpus.treebank.tagged_sents()[22]
sent
```

Out [1]:

```
[('The', 'DT'),
 ('U.S.', 'NNP'),
 ('is', 'VBZ'),
 ('one', 'CD'),
 ('of', 'IN'),
 ('the', 'DT'),
 ('few', 'JJ'),
 ...
]
```

Etiquetado de palabras

REN en NLTK-Python (II)

In [1]:

```
print(nltk.ne_chunk(sent))
```

Etiquetado de palabras

REN en NLTK-Python (II)

In [1]:

```
print(nltk.ne_chunk(sent))
```

Out[1]:

```
(S
  The/DT
  (GPE U.S./NNP)
  is/VBZ
  one/CD
  ....
  according/VBG
  to/TO
  (PERSON Brooke/NNP T./NNP Mossman/NNP)
  ...)
```