

Procesamiento de Lenguaje Natural Pt. 1



Delta Analytics construye capacidad técnica alrededor del mundo.



El contenido de este curso está siendo desarrollado activamente por Delta Analytics, una organización sin fines de lucro 501(c)3 del Área de la Bahía que apunta a capacitar a las comunidades para aprovechar sus datos.

Por favor comuníquese con cualquier pregunta o comentario a inquiry@deltanalytics.org.

Descubre más sobre nuestra misión [aquí](#).

Módulo 6.1:

Procesamiento de Lenguaje Natural



Checklist del Módulo

- ☐ ¿Qué es Procesamiento de Lenguaje Natural y cuáles son sus aplicaciones?
- ☐ ¿Qué es el texto como datos?
 - ☐ Bag-of-words, vectores, matrices
- ☐ Algoritmos de aprendizaje supervisado
 - ☐ Clasificación de texto (Naive Bayes)
- ☐ Próximos pasos para NLP

¿Qué es el procesamiento
del lenguaje natural?



Lenguaje natural vs. lenguaje artificial

El lenguaje natural se define como “un lenguaje que se ha desarrollado naturalmente en uso (en contraste con un lenguaje artificial o un código de computadora).”

The New York Times | <https://nyti.ms/2qXzVH2>

SCIENCE

The Science Behind the Flamingo's One-Legged Stance

Trilobites

By JOANNA KLEIN MAY 24, 2017

Squat down as if you're going to sit in a chair. Make sure to keep your back straight, use your hips as a hinge and push your butt backward. Try not to lean forward. Maintain your knees and ankles at 90-degree angles. Now try it on just one leg, and then swap that one with the other. To make it even harder, stand on a foam mat — and close your eyes.

You may feel your body wobbling, or you may fall over. If only you were a flamingo.

Flamingos can stand on one leg for a really long time. They even do it while sleeping. And according to a study published Tuesday in Biology Letters, flamingos may not even need to use their muscles for the task.

```
def add5(x):  
    return x+5  
  
def dotwrite(ast):  
    nodename = getNodeName()  
    label=symbol.sym_name.get(int(ast[0]),ast[0])  
    print '    %s (label="%s' % (nodename,label),  
    if isinstance(ast[1],str):  
        if ast[1].strip():  
            print '= %s";' % ast[1]  
        else:  
            print '"]'  
    else:  
        print '"]';  
        children = []  
        for n, child in enumerate(ast[1:]):  
            children.append(dotwrite(child))  
        print '    %s -> (' % nodename,  
        for name in children:  
            print '%s' % name,
```



En un área, los humanos aún le ganan a las computadoras:

Lenguaje natural!

Leer, escribir, comunicar y trabajar con datos no estructurados..



¿Por qué es NLP tan difícil?

Todos nuestros ejemplos en el curso han implicado hasta ahora modelar datos estructurados.

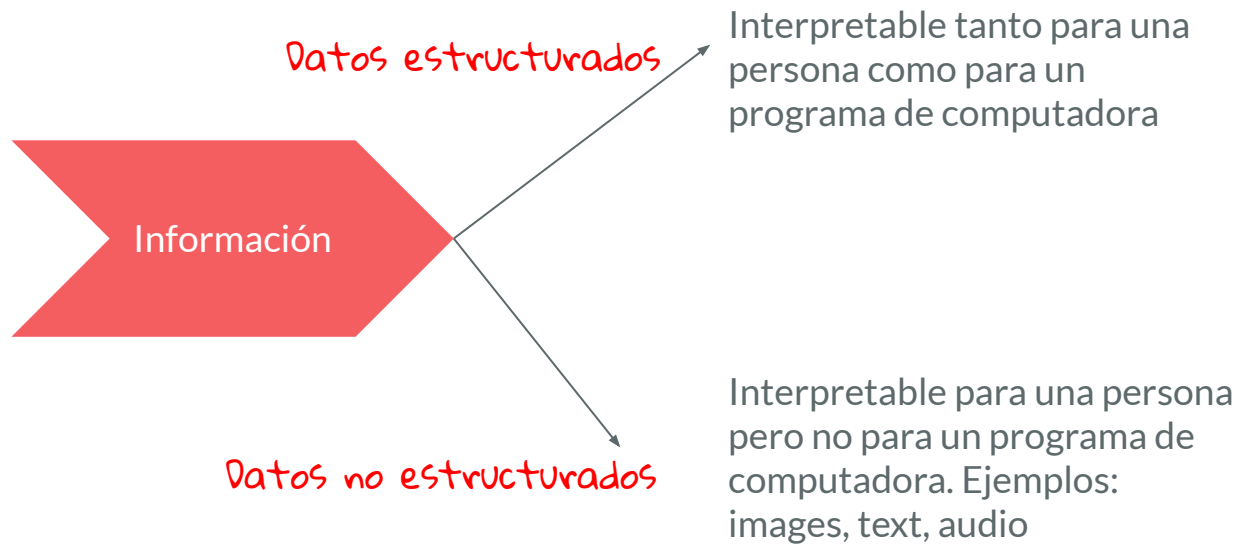
Deuda Pendiente	Ingreso anual (\$)	Aprobación de tarjeta de crédito	Predicción aprobación de tarjeta de crédito
		Y	Y*
200	12,000	No	Yes
60000	60,000	No	No
0	11,000	Yes	No
10000	200,000	Yes	Yes



Datos estructurados (data frame)

NLP es difícil porque el texto es un ejemplo de datos no estructurados.

Nuestros algoritmos saben extraer información desde nuestros datos estructurados pero no son capaces de interpretar datos no estructurados.
¿Cómo extraemos información desde datos no estructurados?



El enfoque en NLP está en cómo representar mejor el lenguaje como estructura de datos!

Representar los datos para retener la mayor cantidad de información posible. Esto no es una tarea trivial

Datos no estructurados

Interpretable para humanos
pero no para programas de
computadora

Representación de datos

Datos estructurados

Interpretable tanto para
humanos como para
programas de computadoras



Música



Imagen



Texto



Ten en cuenta que puedes extraer información aún desde datos no estructurados!

Libros



Labels:

1. Autor
2. Año de publicación
3. Género
4. Número de páginas
5. Título

¿Cómo extraemos información desde datos no estructurados?

¿Por qué más es difícil representar el texto?

Veamos un ejemplo.



Sam envia un texto a Jasmine.



Es tu nombre Wi-fi?
Porque realmente
estoy sintiendo una
conexión

...



Cuál es la diferencia entre cómo un humano y un modelo explicaría este texto?



Es tu nombre Wi-fi?
Porque realmente
estoy sintiendo una
conexión

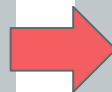
...



Intuición humana



Sam está tratando de
construir su relación
con Jasmine.



NLP

%%\$\$\$\$asds
tasjkshdf adbha
sfsfvdf

A diferencia de todos los modelos que hemos visto anteriormente, las computadoras no pueden entender texto sin formato.

Primero necesitamos representar el texto sin formato de una forma en la que las computadoras lo puedan usar



Para una máquina es difícil representar el lenguaje porque:



Es tu nombre
Wi-fi?
Porque realmente
estoy sintiendo
una conexión

El lenguaje es ambiguo:

- Sutilezas
- Referencias Culturales
- Bromas, sarcasmos, juegos de palabras
- Similitudes fonéticas entre palabras

Existen ciertas reglas gramaticales que podemos aprovechar, pero esto solo nos lleva a la mitad.

Problemos un breve cuestionario que demostrará la ambigüedad lingüística:

El trofeo no cabe dentro del maletín marrón porque es demasiado pequeño. ¿Qué es demasiado pequeño?

¿El maletín o el trofeo?



El trofeo no se ajusta en el maletín marrón porque **es demasiado grande**. ¿Qué es demasiado grande?

¿El maletín o el trofeo?





Podemos fácilmente decir que es "eso", incluso si las oraciones son totalmente diferentes, usando nuestro propio entendimiento de trofeos y maletines.

Una computadora no lo entenderá tan fácil.

¿Si NLP es tan difícil, por qué nos molestamos?



¡Porque NLP es importante!

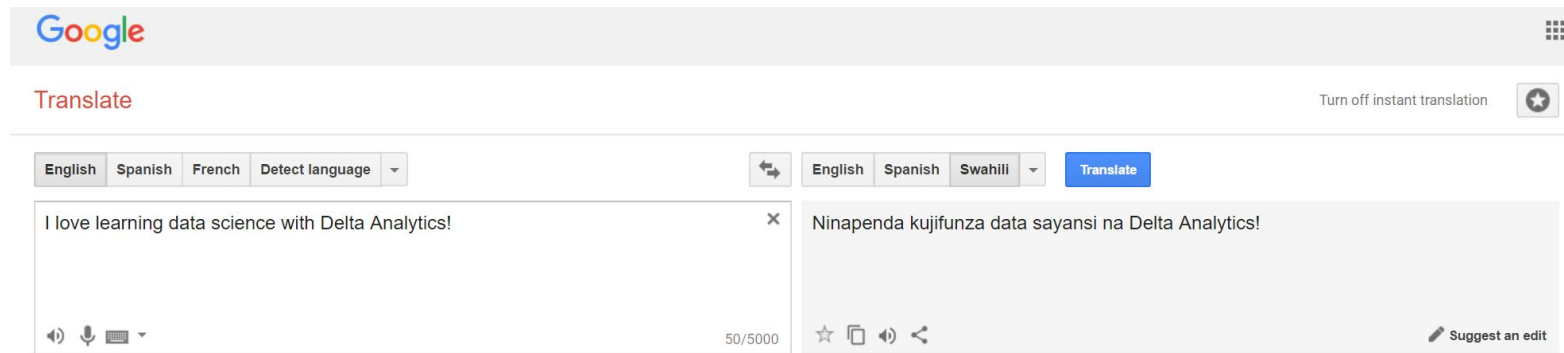
- La mayoría de los datos en el mundo no están representados
- Grandes cantidades de datos están almacenados como texto (**piensa en sitios web, leyes, correos electrónicos, libros**) y esta cantidad aumentará en el futuro
- Permitir que las máquinas entiendan el texto, desbloquea grandes cantidades de información.
¡NLP es increíblemente poderoso!

¡Usas la tecnología NLP todos los días!



Traducción automática & recomendaciones

1. El traductor de Google está impulsado por NLP y aprendizaje profundo



1. New York Times recomienda nuevos artículos basados en la extracción de palabras claves de temas




Fuente: New York Times - [LDA recommendation](#), Google translate, [the great AI awakening](#)



Aplicaciones familiares de NLP

Clasificación de texto: Agregar etiquetas a trabajos académicos



US National Library of Medicine
National Institutes of Health

[Display Settings:](#) ☒ Abstract

[Send to:](#) ☐

Nature, 2014 Mar 20;507(7492):323-8. doi: 10.1038/nature13145. Epub 2014 Mar 12.

Coupling of angiogenesis and osteogenesis by a specific vessel subtype in bone.

Kusumbe AP¹, Ramasamy SK¹, Adams RH².

Author information

Abstract

The mammalian skeletal system harbours a hierarchical system of mesenchymal stem cells, osteoprogenitors and osteoblasts sustaining lifelong bone formation. Osteogenesis is indispensable for the homeostatic renewal of bone as well as regenerative fracture healing, but these processes frequently decline in ageing organisms, leading to loss of bone mass and increased fracture incidence. Evidence indicates that the growth of blood vessels in bone and osteogenesis are coupled, but relatively little is known about the underlying cellular and molecular mechanisms. Here we identify a new capillary subtype in the murine skeletal system with distinct morphological, molecular and functional properties. These vessels are found in specific locations, mediate growth of the bone vasculature, generate distinct metabolic and molecular microenvironments, maintain perivascular osteoprogenitors and couple angiogenesis to osteogenesis. The abundance of these vessels and associated osteoprogenitors was strongly reduced in bone from aged animals, and pharmacological reversal of this decline allowed the restoration of bone mass.

Comment in

Bone biology: Vessels of rejuvenation. [Nature. 2014]

MeSH Terms

[Aging/metabolism](#)

[Aging/pathology](#)

[Animals](#)

[Blood Vessels/anatomy & histology](#)

[Blood Vessels/cytology](#)

[Blood Vessels/growth & development](#)

[Blood Vessels/physiology*](#)

[Bone and Bones/blood supply*](#)

[Bone and Bones/cytology](#)

[Endothelial Cells/metabolism](#)

[Hypoxia-Inducible Factor 1, alpha Subunit/metabolism](#)

[Male](#)

[Mice](#)

[Mice, Inbred C57BL](#)

[Neovascularization, Physiologic/physiology*](#)

[Osteoblasts/cytology](#)

[Osteoblasts/metabolism](#)

[Osteogenesis/physiology*](#)

[Oxygen/metabolism](#)

[Stem Cells/cytology](#)

[Stem Cells/metabolism](#)

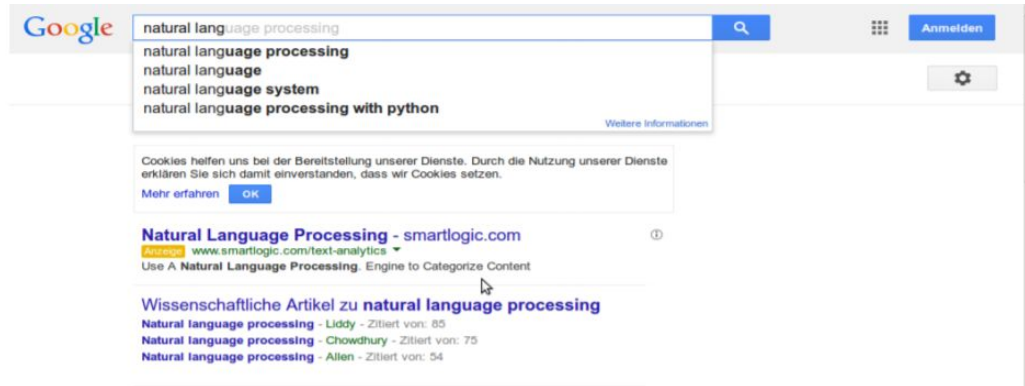


Corrector autográfico, autocompletado y resultados de búsqueda:

Cuando escribes una consulta como '**natural language processing**' en el buscador de Google, ya se están realizando 3 tareas de NLP:

1. Corrector ortográfico y gramatical
2. Autocompletado de resultados de búsqueda
3. Recuperación de información

Todas estas son tareas de NLP!



Asistentes personales de Inteligencia Artificial

**x.ai is a personal assistant who
schedules meetings for you**

Sign up for your free trial

“x.ai es un asistente personal que programa reuniones para ti”

x.ai depende en gran medida de **NLP** para programar reuniones:

1. x.ai **extrae información** desde tus correos electrónicos para agendar:
 - a. Día de la semana
 - b. Ubicación
 - c. Número de personas a agendar
2. *Si respondes a tu asistente no humano (AI), ellos te responden de nuevo! ¿Cómo? x.ai extrae palabras claves desde tu correo electrónico y usa estas palabras para clasificar el texto como que tiene una determinada intención (**clasificador de texto**).*



NLP es a menudo un paso intermedio en procesos complejos



Siri de Apple convierte el sonido de tu voz en palabras para realizar acciones.

Para convertir las palabras en acción, Siri necesita “entender” lo que le estás pidiendo que realice. **Esto requiere NLP!**



Las aplicaciones de NLP son muy variadas! Todas implican primero representar el texto como datos, y luego extraer el significado del texto

Aplicaciones del NLP

- ✓ Clasificadores - Filtros de Spam
- ✓ Recuperación de información - Buscador de Google
- ✓ Extracción de información - AI asistente
- ✓ Máquina traductora - Google translate
- ✓ Preguntas & Respuestas - Respuesta automática a ciertos e- mails
- ✓ Resumen de texto - ¿De qué se trata este artículo?
- ✓ Análisis de sentimientos - ¿es este documento positivo o negativo?
- ✓ Procesamiento del habla - Opciones en las líneas telefónica de ayuda
- ✓ Reconocimiento óptico de caracteres - Escáner de cheques en cajeros automáticos
- ✓ Corrector ortográfico, autocompletado - Buscador de Google



¿Cómo convertimos el texto
en datos?



¡Representar el texto como datos es una gran parte de nuestro trabajo de NLP!



Lo que hace que el NLP sea diferente de los otros algoritmos que hemos estudiado hasta ahora, es lo **difícil** y lo **computacionalmente intenso** que puede ser este paso.



Representar
texto como
dato

Limpieza de datos

stemming

lemmatization

Stop words

Limpieza adicional

Representación de
los datos

tokenizing

n-gramas

bag-of-words

No necesariamente
en este orden -
depende de tu
pregunta de
investigación

Primero, limpiamos el
texto de entrada y
luego lo representamos
usando un modelo.

Veamos cada una de estas
técnicas
¡Paso a paso!



Recordando el ejemplo: Sam envía un texto a Jasmine.



Es tu nombre Wi-fi?
Porque realmente
estoy sintiendo una
conexión

...



Primero, algo de vocabulario!



En nuestro intercambio de texto entre Sam y Jasmine:



Característica



Palabra



Texto
Documento



Todos los textos
Corpus

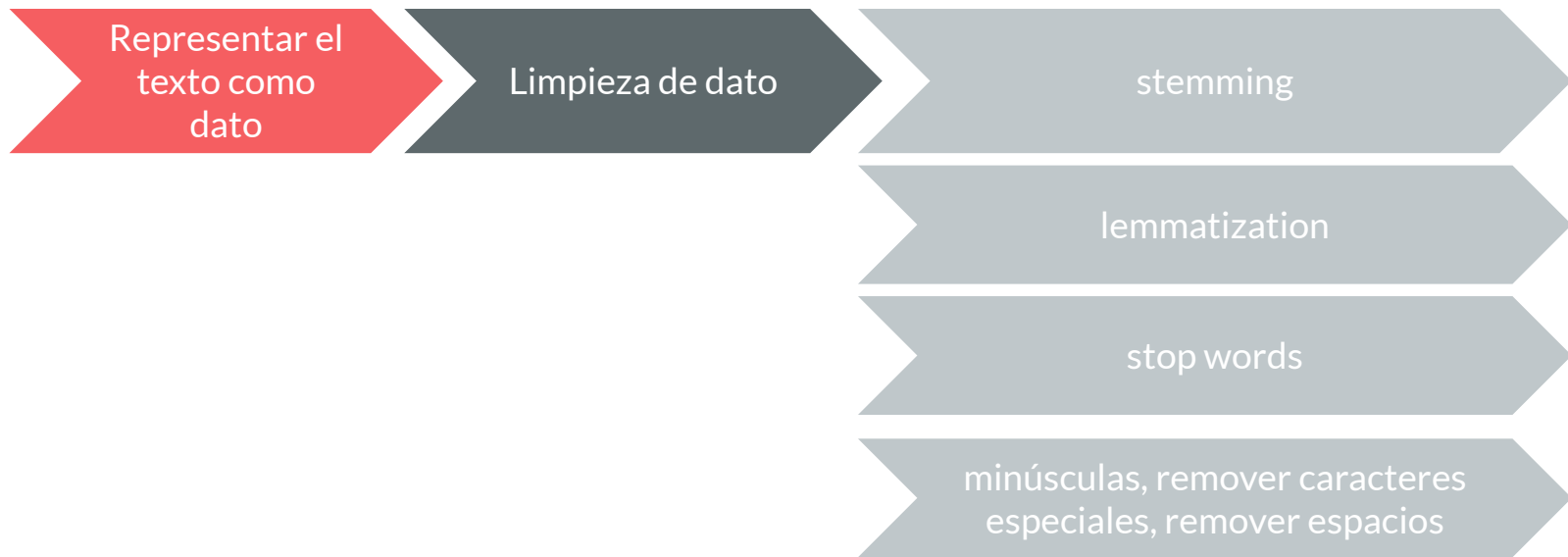
Observación

Conjunto de datos

Comencemos con la
limpieza de texto:



El objetivo de limpiar los datos de textos es estandarizar el texto lo mayormente posible, de modo que las palabras con la misma intención se agrupen.



Representar
texto como
dato

Limpieza
de datos

Stemming

Reducción de palabras a su forma raíz. Ej. "Explorando -> "Explorar"



Oración original:

"Hola Jasmine, ¿te gustaría que salgamos este fin de semana? Podríamos ver una película y cenar después! "

Palabras a stem (tallo):

"salgamos"" → "salir"



Representar
texto como
dato

Limpieza
de datos

Lematización

Lematización es similar a stemming, pero utiliza el análisis morfológico de las palabras para volverlas a su raíz.



Original	Stemmed	Lematizado
walked	walk	walk
are	are	be
natural	natur	natural
processing	process	process



Representar
texto como
dato

Limpieza
de datos

Stop words

Al eliminar "stop words", se eliminan las palabras comunes que no agregan significado



Oración original:

"Hi Jasmine, would you like to go running in the park this weekend? We could catch a movie and dinner afterwards! "

Oración con stopwords removidas:

"Hi Jasmine, would you like to go running in the park this weekend? We could catch a movie and dinner afterwards! "



Stop words

Dos principales formas de abordar la eliminación de stop words



Lista
predeterminada
de palabras

Usando una lista de stopwords predeterminada (disponible a través del paquete nltk).

adaptada desde
el corpus

Descartar palabras que aparecen con demasiada frecuencia en tus datos (usando conteos de frecuencia)

Representar
texto como
dato

Limpieza
de datos

Limpieza
adicional

Minúsculas, eliminar espacios,
eliminar caracteres especiales
(%,?!,)

Letras minúsculas

Eliminar caracteres
especiales

Oración original:

“Hola Jasmine, ¿te gustaría salir a correr en el parque este fin de semana? Podríamos ver una película y cenar después!”

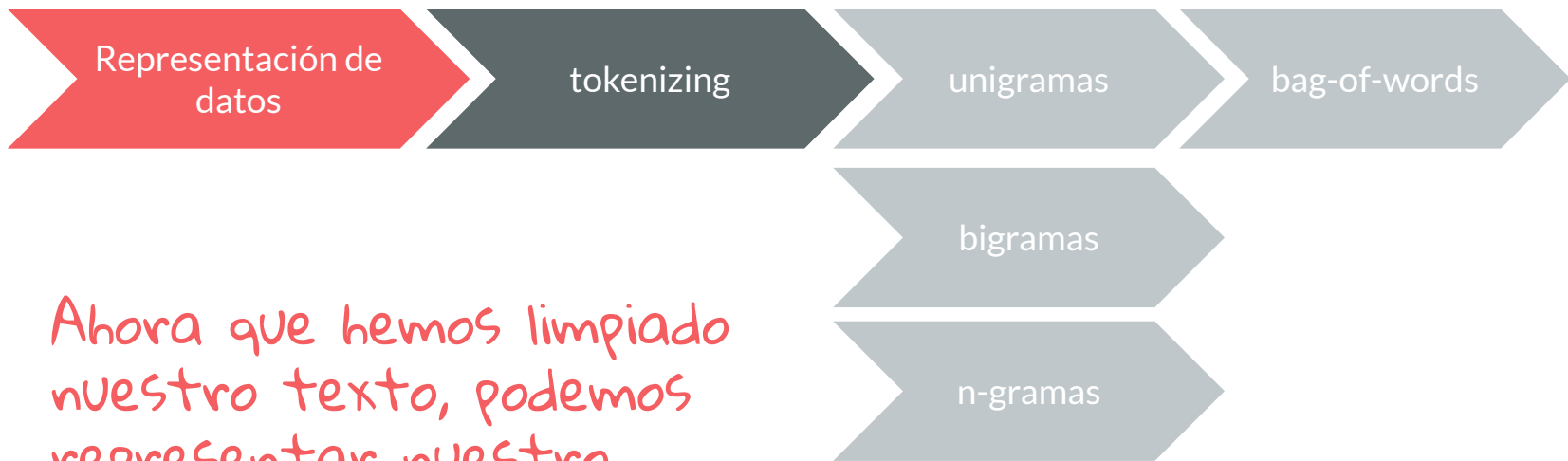
Oración limpia:

“hola jasmine te gustaría salir a correr en el parque este fin de semana podríamos ver una película y cenar después”



Una vez que el texto es limpiado,
el siguiente paso es *tokenización*.





Ahora que hemos limpiado nuestro texto, podemos representar nuestro texto como datos usando un proceso llamado tokenización.

Veamos!

Representación
de datos

tokenizing

Desarmando el texto en palabras,
frases, elementos (o "tokens")



Oración original:

"Hi Jasmine, would you like to go running in the park this weekend? We could catch a movie and dinner afterwards! "

Oración tokenizada:

"Hi", "Jasmine", "would", "you", "like", "to",
"go", "running", "in", "the", "park", "this",
"weekend", "We", "could", "catch", "a", "movie"
"and", "dinner," "afterwards"



Tokenizing

Puedes tokenizar a unigramas, bigramas,...
n-gramas. Nuestros algoritmos usarán unigramas.



Unigrama
Una palabra

“Hi”, “Jasmine”, “would”, “you”,
“like”, “to”, “go”, “running”, “in”, “the”,
“park”, “this”, “weekend”,

Bigrama

Dos palabras

“Hi Jasmine”, “would you”, “like
to”, “go running”, “in the”, “park
this”, “weekend”,

n-grama
n palabras

“Hi Jasmine would you”, “like to
go running”, “in the park this”,
“weekend”,



¿Existe un **mejor** tipo de tokenización? Como muchas de nuestras opciones en ML, hay un termino medio

- Los modelos con unigramas ignoran el contexto de una palabra, por lo que el sesgo es alto.
- Los modelos con bigramas y trigramas toman en cuenta el contexto, pero necesitan más datos para entrenar.



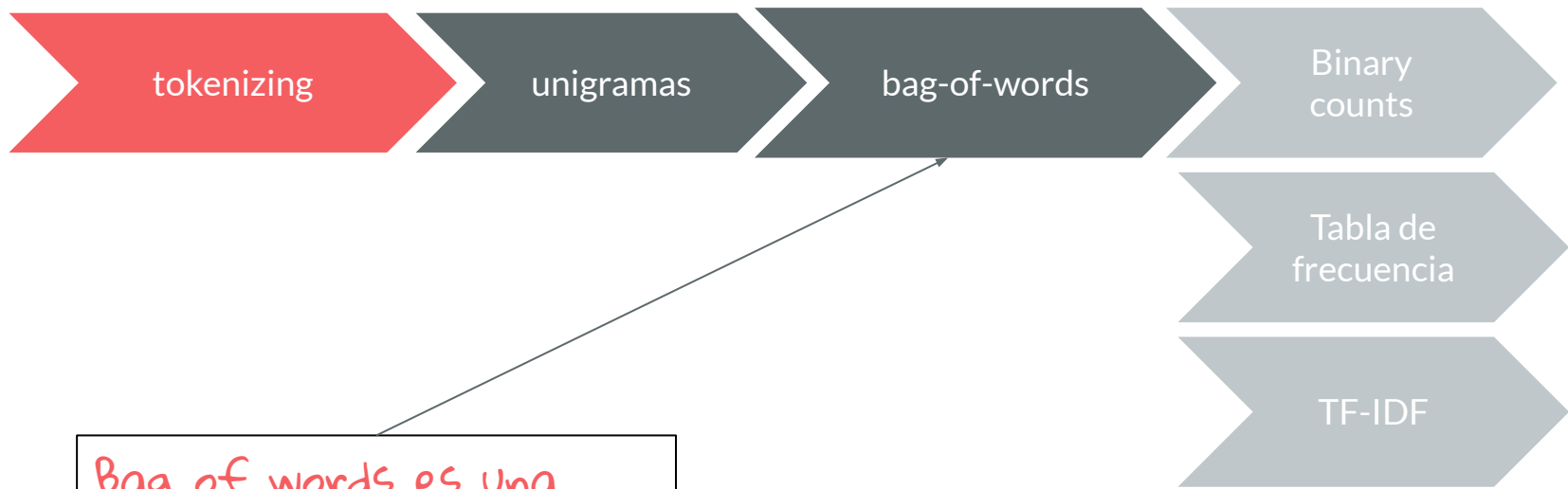
Los algoritmos de bag-of-words usan tokenización de unigrama, por lo que su desventaja clave es que ellos no toman en cuenta el contexto de ninguna palabra.

Piénsalo: con bag-of-words, un documento que dice “es malo, no es bueno” se consideraría igual que un documento que dice “es bueno, no es malo”.



Veamos más de cerca la
representación de
Bag-of-Words.





Bag of words es una representación de texto que usa tokenización de unigrama.

Después de la tokenización, las palabras se pueden representar de diferentes maneras.

Introduciremos conteos binarios, tabla de frecuencia, TF-IDF.



Tokenizing

bag-of-words

Bag of words usa tokenización de unigrama en el corpus of mensajes de texto!

Texto 1:

“Hi Jasmine, would you like to go running in the park this weekend? We could catch a movie and dinner afterwards! ”

Texto 2:

“Great dinner Jasmine! Sweet dreams.”

Texto 1:

“Hi”, “Jasmine”, “would”, “you”, “like”, “to”, “go”, “running”, “in”, “the”, “park”, “this”, “weekend”, “We”, “could”, “go”, “to”, “a”, “movie” “and”, “dinner,” “afterwards”

Texto 2:

“Great”, “dinner”, “Jasmine”, “Sweet”, “dreams”



documento

corpus

Texto único → Textos múltiples



El primer paso en la representación de la representación de bag-of-word es la tokenización en unigramas de cada documento. Luego, estas palabras se agregan utilizando diferentes enfoques.

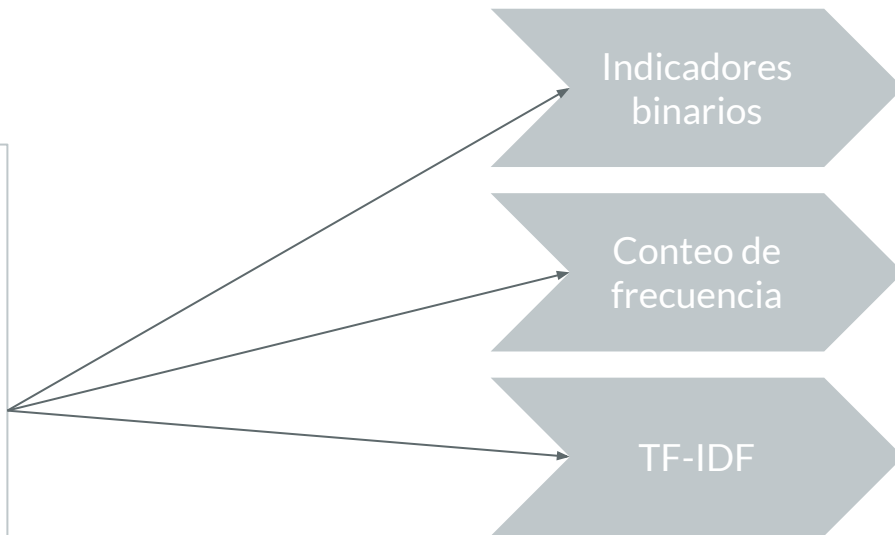
1) Tokenización en unigramas de cada documento

Texto 1:

"Hi", "Jasmine", "would", "you", "like", "to", "go", "running", "in", "the", "park", "this", "weekend", "We", "could", "go", "to", "a", "movie" "and", "dinner," "afterwards"

Texto 2:

"Great", "dinner", "Jasmine", "Sweet", "dreams"



TF-IDF, conteos binarios y de frecuencia son enfoques de agregación muy simples.



bag-of-words

Conteos
binarios

Los indicadores Binarios son 1 o 0 dependiendo si la palabra está presente en el documento o no.

Texto 1:

“Hi”, “Jasmine”, “would”, “you”, “like”, “to”,
“go”, “running”, “in”, “the”, “park”, “this”,
“weekend”, “We”, “could”, “go”, “to”, “a”,
“movie” “and”, “dinner”, “afterwards”

Texto 2:

“Great”, “dinner”, “Jasmine”, “Sweet”,
“dreams”

texto	Jasmine	dinner	dreams	weekend	...
1	1	1	0	1	...
2	1	1	1	0	...



Bag-of-words

Conteo de
frecuencia

Conteos de frecuencia son el número de veces que cada palabra aparece en el documento

Texto 1:

"Hi", "Jasmine", "would", "you", "like", "to",
"go", "running", "in", "the", "park", "this",
"weekend", "We", "could", "go", "to", "a",
"movie" "and", "dinner," "afterwards"

Texto 3:

"Hi", "Jasmine", "how", "is", "your", "week",
"going" "I" "am" "going" "bowling" "this"
"saturday" "want" "to" "come"

text	Jasmine	dinner	go	going	...
1	1	1	2	0	...
3	1	0	0	2	...

Problema: ¿Qué tan importante es cada palabra?



Bag-of-words

Conteo de
frecuencia

Problema: ¿Qué tan importante es cada palabra?

Texto 1:

“Hi”, “Jasmine”, “would”, “you”, “like”, “to”,
“go”, “running”, “in”, “the”, “park”, “this”,
“weekend”, “We”, “could”, “go”, “to”, “a”,
“movie” “and”, “dinner”, “afterwards”

Texto 3:

“Hi”, “Jasmine”, “how”, “is”, “your”, “week”,
“going” “I” “am” “going” “bowling” “this”
“saturday” “want” “to” “come”

text	Jasmine	dinner	go	going	...
1	1	1	2	0	...
3	1	0	0	2	...

La frecuencia nos dice cuán importante es una palabra para el documento. *Pero ¿cómo podemos juzgar cuán importante es esa palabra en el corpus?* Ej. “Jasmine” probablemente tiene una frecuencia alta, pero esto no nos da tanta información nueva - ¡Todo el corpus es sobre ella!
Tf-idf ayuda a determinar la importancia en relación con el corpus.





Bag-of-words

TF-IDF

TF-IDF: Frecuencia del término,
Frecuencia de documento inversa

Tf-idf es un número asignado a cada palabra para reflejar **cuán importante es esa palabra para un documento**.

Tf-idf **aumenta** proporcionalmente el número de veces que aparece una palabra en el documento, pero se **compensa** con la frecuencia de la palabra en el corpus

Procedimiento de 3 pasos:

- 1) Calcular frecuencia de término
- 2) Calcular la frecuencia de documento inversa
- 3) Multiplicar 1)*2)



Bag-of-words

TF-IDF

TF-IDF vs. frecuencia

TF-IDF puede decirnos qué palabras proporcionan la mayor información sobre un documento determinado. Comparemos la frecuencia y tf-idf:

Frecuencia

	Jasmine	Sam	running
Text 1	1	2	0
Text 2	1	1	0
Text 3	2	1	1

Tf-idf

	Jasmine	Sam	running
Text 1	0.2	0.1	0
Text 2	0.01	0.02	0
Text 3	0.1	0.1	0.8

Basados solo en las frecuencias, vemos que las palabras “Jasmine” y “Sam” son importantes. Pero lo sabíamos! La tabla tf-idf proporciona un poco más de información: La palabra “running” es **relativamente más importante**. Si nos basáramos sólo en las frecuencias, podríamos haber perdido la importancia relativa de la palabra “running”!

Veamos cómo calcular tf-idf a continuación.



Frecuencia del
término

Calculado para cada palabra en un documento.
Igual al recuento de frecuencia normalizado para
tener en cuenta la longitud del documento.

TF: Frecuencia del término: mide con
qué frecuencia se produce un término
en un documento.

$TF(t) = (\text{Número de veces que el término } t \text{ aparece en un documento}) / (\text{Número total de términos en el documento})$.

Texto 1:

“Hi”, “Jasmine”, “would”, “you”, “like”, “to”,
“go”, “running”, “in”, “the”, “park”, “this”,
“weekend”, “We”, “could”, “go”, “to”, “a”,
“movie” “and”, “dinner,” “afterwards”

Tabla de frecuencia de términos:

	running	Total words in text	TF(T)
Text 1	1	22	0.04

La frecuencia de la palabra running
en el texto 1 es 0.04 (1/22).



Computed across every single
document in the corpus

Inverse Document Frequency measures how important a term is.

While computing TF, all terms are considered equally important.

However, certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance.

$IDF(t) = \log_e(\text{Total number of documents} / \text{Number of documents with term } t \text{ in it})$

TF	IDF
All terms considered equally important	Weights down frequent term while scales up infrequent terms.
'Is' as important as 'girlfriend'	"Vacation" will be more important than "and"

Frecuencia de
documento
inversa

$IDF(t) = \log_e(\text{Número total de documentos} / \text{Número de documentos con el término } t)$

Texto 1:

"Hi", "Jasmine", "would", "you", "like", "to",
"go", "running", "in", "the", "park", "this",
"weekend", "We", "could", "go", "to", "a",
"movie" "and", "dinner," "afterwards"

Texto 2:

"Great", "dinner", "Jasmine", "Sweet",
"dreams"

Numero total de documentos	running	IDF(T)
2	1	0.69


$$IDF(\text{running}) = \log_e(2/1)$$



Bag-of-words

TF-IDF

Unámoslo todo!!

Procedimiento de 3 pasos:

- 1) Calcular frecuencia de término
- 2) Calcular la frecuencia de documento inversa
- 3) Multiplicar 1)*2)

	palabra	TF(T)	IDF(T)	TF-IDF(T)
Text 1	running	0.04	0.69	0.0276

TF-IDF nos dice qué palabras son importantes y cuáles no, en **relación con el corpus!**

Por lo tanto, la composición del corpus también puede ser una importante decisión como investigador



Binario, Frecuencia y TF-IDF todo resulta en matrices dispersas. Una matriz dispersa es un marco de datos donde la mayoría de los elementos=0.

La matriz es “**dispersa**” porque cada característica es una palabra en el corpus. Para un gran corpus, esta será un extenso vocabulario. *La mayoría de los elementos en la matrix serán 0.*

text	Jasmine	dinner	dreams	weekend
1	1	1	0	1
2	1	1	1	0
3	1	0	0	0

En nuestro ejemplo, nuestro corpus es el conjunto de mensajes de texto.

Text 3:

“Hey”, “Jasmine”, “how”, “is”, “your”, “week”, “going”



Entonces, dónde estamos?



Pusimos mucho esfuerzo en limpiar nuestros datos y exploramos un simple modelo (bag-of-words) para representar nuestro texto.

Ahora, finalmente podemos comenzar a realizar un análisis exploratorio.



Análisis exploratorio

Con paquete “nltk” de Python



Análisis exploratorio

Frecuencia

Cuenta las veces que cada palabra aparece en el corpus

Frecuencia
condicional

Cuenta la cantidad de veces que cada palabra aparece en el corpus dependiendo de un criterio adicional

Concordancia

Una lista que muestra todas las apariciones de un texto de frase

similar

Devuelve palabras en un rango de contextos similares al de la palabra de entrada

collocations

Secuencias de palabras que generalmente aparecen juntas

Análisis
exploratorio

Frecuencia

La frecuencia cuenta la cantidad de veces que cada palabra aparece en el corpus



Frecuencia de la palabra “amor” en textos entre Sam y Jasmine a lo largo del tiempo



Fecha de inicio de la relación

`nltk.ConditionalFreqDist()`



Cuenta la cantidad de veces que cada palabra aparece en el corpus según criterios adicionales

Por ejemplo, es posible que deseemos ver la frecuencia de las palabras por tipo de artículo de noticias. Un artículo está categorizado como “noticia” y el otro como romance “romance.”

Condition: News

the	
cute	
Monday	
could	
will	

Condition: Romance

the	
cute	
Monday	
could	
will	

Análisis
exploratorio

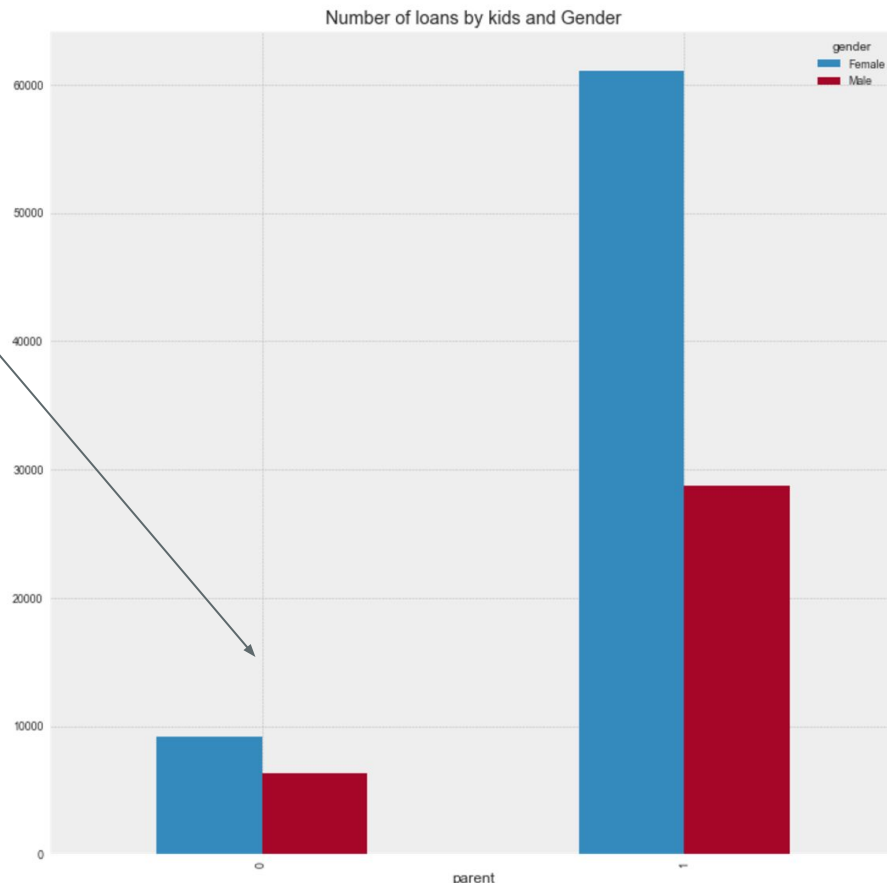
Frecuencia
condicional

¡Ya has visto un ejemplo de frecuencia condicional en nuestros datos KIVA!!



Este es un ejemplo de frecuencia condicional de una coincidencia de palabras por género

Campo “Niños” definido como 1 para cualquier descripción que tenga las palabras “chicos”, “niño”, “niña”, “hija”, “hijo”, “madre”, “padre”.



Una lista que muestra el contexto que rodea una palabra o frase de entrada.

Todas las apariciones de la palabra “mujer” en los datos de Kiva.

```
In [74]: text_corpus.concordance('woman')
```

Displaying 25 of 237 matches:

filling life dorine is a 27-year-old	woman	with three kids all of whom attend s
aid successfully habiba is a married	woman	with 4 children ranging in age from
ocation in mombasa asha is a married	woman	with four children all of whom atten
the loan promptly nzije is a married	woman	with three children all of whom atte
thful 28-year-old mrembo (beautiful	woman	who has been blessed with five schoo
“ she said millicent mobilized other	woman) she is a wise farmer whom neighbor
ease her income level millicent is a	woman	in her village and joined juhudi kil
le within 5 years grace is a married	woman	who is making a change for herself a
cess loans from banks since she is a	woman	and owns a house that has piped wate
ical decisions margaret is a married	woman	and a smallholder farmer she joined
e solar lights mwanasha is a married	woman	with two children both of whom atten
n shillings) to buy poultry being a	woman	with three children all of whom atte
ends her dreams to be an independent	woman	accessing funds is very challenging
microfinance bank rose is a married	woman	and to have job security can now com
live happily mwanamgeni is a married	woman	with one child who attends school sh
	woman	with four kids all of whom attend sc

Collocations son secuencias de palabras que generalmente aparecen juntas

```
In [76]: text_corpus.collocations()
```

```
years old; acre fund; one acre; juhudi kilimo; piped water; join yehu;  
greatest monthly; school fees; major challenge; primary customers;  
married woman; access loans; kadet ltd; first loan; attend school;  
solar lights; wheat flour; microfinance bank; monthly expense; three  
children
```

Las Collocations aprovechan el hecho de que existan pares de palabras que naturalmente van juntas. Esto suele ser particular del conjunto de datos, por ejemplo **Acre Fund** es el nombre de un socio y es poco probable que sea una collocation en un corpus más amplio.

"Similar" en NLTK toma las palabras que aparecen alrededor de nuestra palabra de entrada como contexto, y encuentra otras palabras que tienen contexto similar.

El resultado es una lista de palabras similares a nuestra palabra de entrada!

```
In [35]: text_corpus.similar("mother")
```

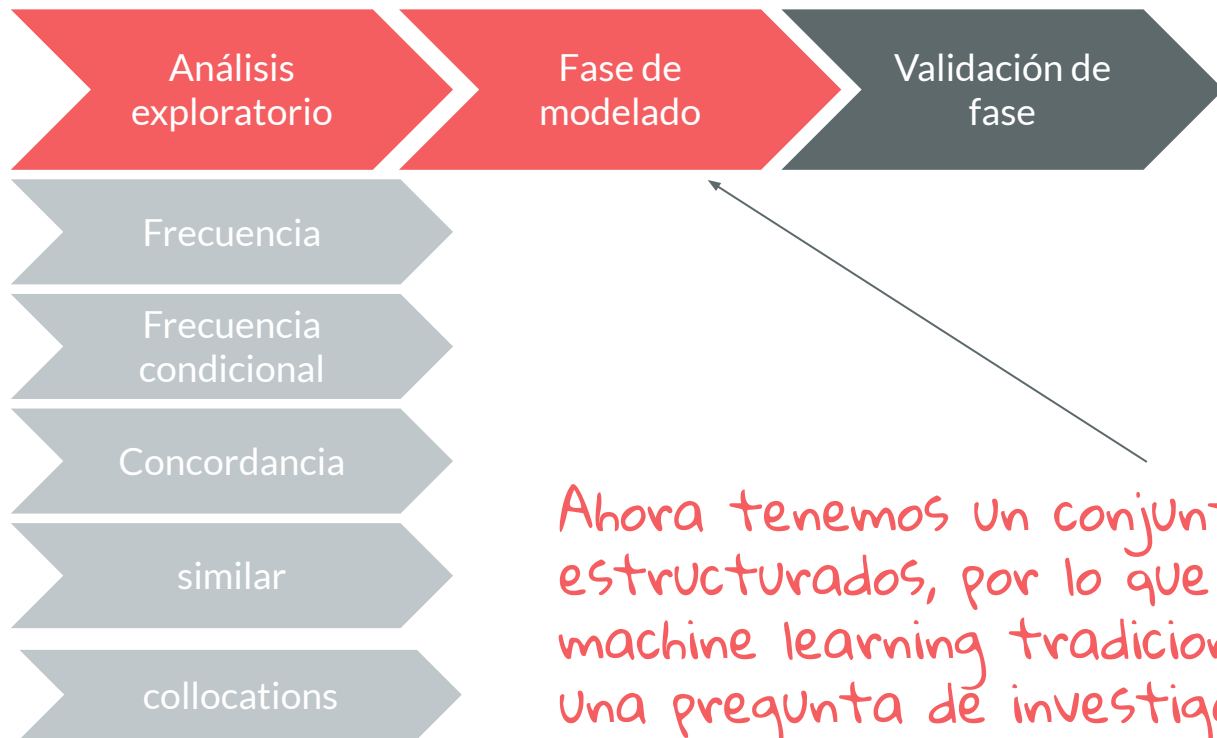
```
woman father business loan farmer man house lady and challenge married  
that children friend profit lack living hope life sale
```

```
In [38]: text_corpus.similar("father")
```

```
mother woman challenge business loan lack farmer way house women  
variety hope life sale stock number lives lady part group
```

Fuente: Sample of Kiva data

Ahora que tenemos una idea de lo que hay en los datos, pasemos a nuestra fase de modelado.

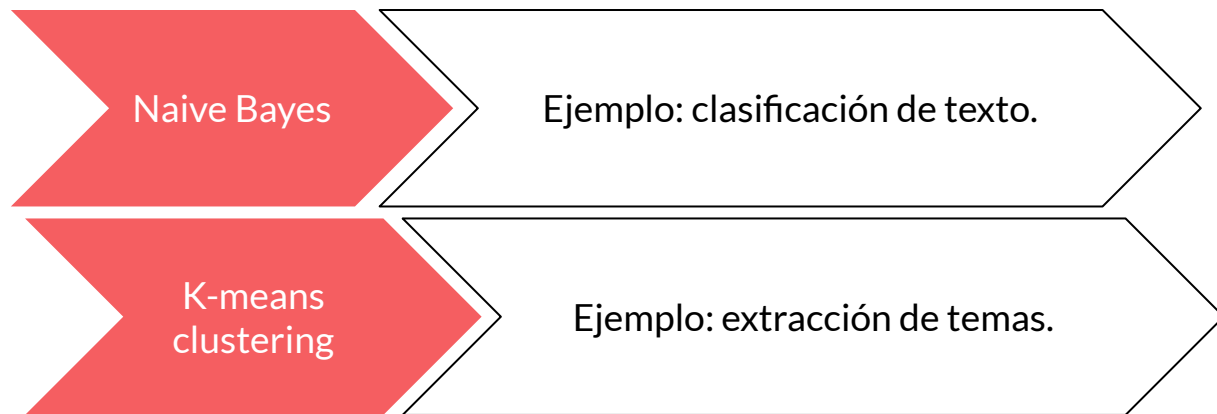


Ahora tenemos un conjunto de datos estructurados, por lo que podemos utilizar machine learning tradicional para responder una pregunta de investigación.

Modelado de NLP

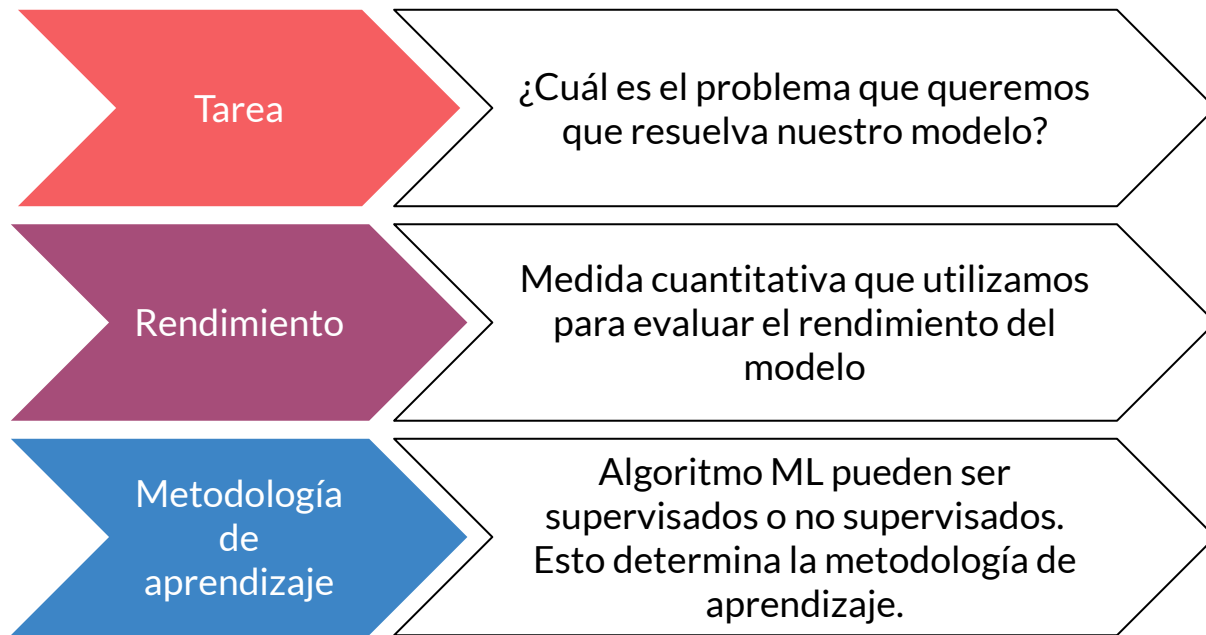


Discutiremos dos algoritmos:



Este es solo un punto de partida para tu exploración de NLP. Ambos algoritmos son bastante intuitivos y, para algunos casos de uso, producen resultados poderosos.

Utilizaremos nuestro familiar framework para analizar tanto naive bayes como k-means clustering:



¿Recuerdas la diferencia entre supervisado y no supervisado?



Algoritmos Supervisados

- Cada observación tiene una etiqueta asociada (para cada x , hay una Y verdadera)
- El objetivo es predecir Y usando x .
- Como tenemos la verdadera Y , podemos decir qué tan lejos está Y^* de la verdadera.
- **La mayoría del aprendizaje automático está supervisado.**



Algoritmos No Supervisados

- Para cada x , no hay una Y .
- Nosotros no sabemos las respuestas correctas.
- En cambio, tratamos de modelar nuestra comprensión de la distribución de x para hacer inferencias sobre Y .

Hoy veremos los modelos PNL supervisados y los no supervisados.

K-means clustering



Sin datos etiquetados

No Supervisado

Algoritmo
ML

Naive Bayes



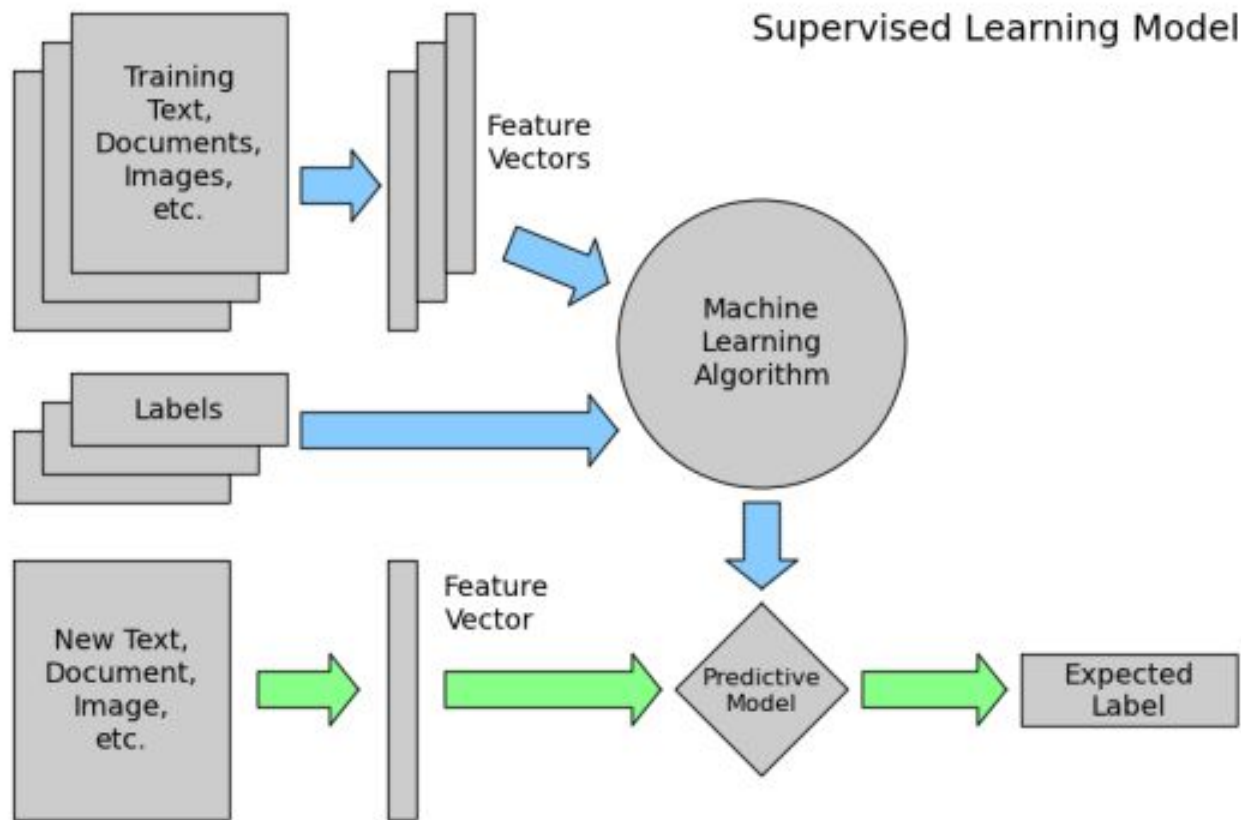
Datos etiquetados Supervisado

1) Naive Bayes:

Algoritmo de clasificación de texto



Algoritmos de aprendizaje supervisado de NLP



Naive Bayes:

Pros

- Simple pero con buen rendimiento, especialmente en la clasificación de documentos.
- Permite que cada característica o palabra en un documento contribuya a su clasificación.
- Baja tasa de falsos positivos.

Constras

- “Naive” - hace suposiciones que no necesariamente se mantienen en la vida real
- Asume independencia entre las características



Tarea

¿Podemos decir si un correo electrónico es spam o no? (Este es un correo electrónico real de una bandeja de entrada!!)

FBI INVESTIGATION ON YOUR FUND IN AFRICA - Federal Bureau Of Investigations Headquarters Washington Dc. Building 935 Pennsylvania Ave. NW WAS

Federal Bureau Of Investigations
Headquarters Washington Dc.
Building 935 Pennsylvania Ave.
NW WASHINGTON, D.C. 20535-0001
E-Mail: fbi_govt2@usa.com

NOTICE OF ONGOING INVESTIGATION

Attn Recipient:

This is Agent Dean Marugor, we were sent by the acting Director of Federal Bureau of Investigation (ANDREW G. McCABE), we are currently in Africa as an FBI/ United States delegate that have been delegated to investigate these fraudsters who are in the business of swindling Foreigners that has transactions in Africa.

Be informed that during our investigations we found out that there is a total amount of \$4.5 Million that has been assigned in your name as the beneficiary and these fraudsters are busy swindling you without any hope of receiving your fund, these are the works of the fraudsters who needed to extort money from you in the name of this transfer, We have to inform you that we have arrested some men in respect of this delayed overdue fund, We have a very limited time to stay in Africa here so I advise you urgently respond to this message.

These criminals will be caught unaware and we don't want them to know this new development to avoid jeopardizing our investigation, you need to conceal anything that has to do with this exercise to enable us get all the necessary information we required.

I will be expecting your swift response as soon as you receive this email and notify us of any message or phone call you receive from those fraudsters for us to investigate on it before you make any contact with them.

In case if found this message in spam folder, it could be due to your Internet Service Provider, ISP. So kindly move to your inbox before replying.

Regards,
Agent Dean Marugor

Email: fbi_govt2@usa.com
Email: deanmarugor2@gmail.com

Federal Bureau of Investigation
+1(205)340-5968



Tarea

¿Cuál es la diferencia entre cómo un humano y un modelo de machine learning decidirían si esto es spam o no?



Texto del email:

*“Tenga en cuenta que durante nuestras investigaciones descubrimos que hay un monto total de **\$4.5 millones** asignado a su nombre como beneficiario y que estos estafadores están tratando de estafarlo sin ninguna esperanza de recibir su fondo.”*

Intuición Humana



- No tengo un fondo en África.
- ¿Me enviaría un correo electrónico el FBI?
- ¿Cómo no sabría acerca de \$ 4.5 millones de dólares?



Naive Bayes

Cada palabra en el correo electrónico contribuye a la probabilidad de que el correo electrónico sea spam, según la frecuencia histórica en los mensajes de spam.

- Mismo objetivo! Tanto los humanos como ML están evaluando la probabilidad de que el correo electrónico sea verdadero.
- Diferencia clave: Naive Bayes usan la frecuencia de las palabras presentes en el correo electrónico en ejemplos históricos de spam para asignar una probabilidad de que sea spam.



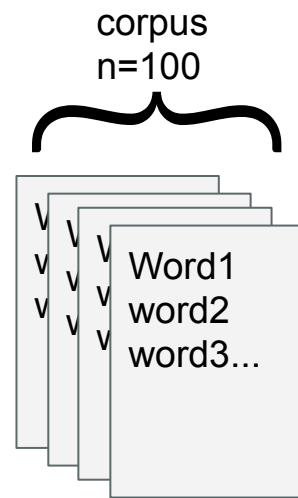
Entrenamos nuestro modelo evaluando cómo cada palabra del correo electrónico **contribuye** a una clasificación de Spam o No Spam, en toda nuestra muestra etiquetada de correos electrónicos.



Calcular la probabilidad de spam **para cada palabra en un email**

Multiplica las probabilidades en todas las palabras.

Si la probabilidad total es > 0.5 , la **clasificamos como spam**.



Repite para cada email en el corpus de datos etiquetados

Task

Predecir si un correo electrónico es spam es un ejemplo de una tarea de clasificación.



Clasificación

Email	"millones"	"FBI"	spam
1	1	1	1
2	0	0	0
3	1	0	0
4	0	0	1
5	1	0	1
6	1	1	1

IMPORTANTE: Tener en cuenta que nuestro data frame es un conteo de bag-of-words. Hicimos tokenización de unigrama para transformar datos no estructurados en datos estructurados.



Lo que queremos es la probabilidad de que un correo electrónico sea spam, DADO las palabras que están presentes en el correo electrónico.

Esta es la probabilidad condicional.





TEXTO POR EMAIL: “Descubrimos que hay un monto total de \$ 4.5 millones asignado a su nombre como beneficiario y estos estafadores están tratando de engañarlo para que no pueda recibir su fondo...”

Formalizado como una ecuación, nuestra probabilidad condicional para el correo electrónico de muestra anterior es la siguiente:

$P(\text{Spam} \mid \text{Descubrimos que hay un monto total de \$4.5 millones...})$

La probabilidad condicional es un concepto que se puede ver en nuestra vida diaria. Por ejemplo:

$$P(\text{Lluvia}|\text{Abril})$$

Probablemente varía bastante, dependiendo de en qué parte del mundo te encuentres

Esta ecuación formaliza la probabilidad de que llueva hoy, dado que es abril.

Para obtener una introducción a este concepto, consulte:

<https://www.khanacademy.org/math/statistics-probability/probability-library/conditional-probability-independence/v/calculating-conditional-probability>



El teorema de Bayes nos permite calcular la probabilidad condicional:

$$\boxed{P(\text{Rain}|\text{April})} = \frac{P(\text{April}|\text{Rain}) * P(\text{Rain})}{P(\text{April})}$$

El teorema de Bayes nos permite calcular la probabilidad condicional:

<https://www.khanacademy.org/math/statistics-probability/probability-library/conditional-probability-independence/v/calculating-conditional-probability>



Tarea

Definiendo
 $f(x)$

¿Cómo podemos usar palabras para predecir el spam?



El algoritmo Naive Bayes calculará la **probabilidad condicional de que un email sea spam**, condicionado al texto del correo electrónico.

Usando nuestra representación de bag-of-words, evaluamos la probabilidad de cada palabra por separado y las resumimos.

$P(\text{Spam} \mid \text{We found out that there is a total amount of \$4.5 million...})$

$= P(\text{Spam} \mid \text{we}) * P(\text{Spam} \mid \text{found}) * P(\text{Spam} \mid \text{total}) * P(\text{Spam} \mid \text{amount}) * P(\text{Spam} \mid \text{million}) \dots$

NOTE: Why did we leave out words like “that”, “is”, “a”, and “of”? These are examples of **stopwords** that would have been removed while we were cleaning and preparing our data.



Tarea

Definiendo
 $f(x)$

Cada palabra en un correo electrónico contribuye a la probabilidad de spam.



Usemos el teorema de Bayes para calcular las probabilidades a continuación, comenzando con $P(\text{Spam}|\text{million})$. Esto nos dirá cuán **predictiva** es la palabra "millones" para determinar el spam.

$P(\text{Spam} \mid \text{We found out that there is a total amount of \$4.5 million...})$

$= P(\text{Spam}|\text{we}) * P(\text{Spam}|\text{found}) * P(\text{Spam}|\text{total}) * P(\text{Spam}|\text{amount}) * P(\text{Spam}|\text{million}) \dots$

NOTA: ¿Por qué omitimos palabras como "that", "is", "a", y "of"? Estos son ejemplos de **stopwords** que se habrían eliminado mientras estábamos limpiando y preparando nuestros datos.



Usemos el Teorema de Bayes para calcular esta probabilidad:

$$\boxed{P(\text{Spam}|\text{million})} = \frac{P(\text{million}|\text{Spam}) * P(\text{Spam})}{P(\text{million})}$$

No sabemos la probabilidad de spam dado "million", pero gracias a nuestros datos de entrenamiento etiquetados, sabemos la probabilidad de que "millones" dado spam.

Calcularemos cada una de los 3 componentes de la derecha, comenzando con $P(\text{million}|\text{Spam})$



$P(\text{million}|\text{Spam})$

¿Cuál es la probabilidad de que "millions" ocurra en un email spam?

¿Cuántos correos electrónicos no deseados en nuestro conjunto de datos de entrenamiento contienen la palabra "millions"?



Probabilidad de que la palabra "millions" esté en un correo electrónico no deseado

$$= \frac{\text{sum}(\text{millions}=1 \ \& \ \text{spam}=1)}{\text{sum}(\text{spam}=1)}$$

$$P(\text{million}|\text{Spam}) = \frac{3}{4}$$

Email	millions	spam
1	1	1
2	0	0
3	1	0
4	0	1
5	1	1
6	1	1

P(Spam)

¿Cuál es la probabilidad de que un correo electrónico sea spam?

¿Cuántos correos electrónicos son spam?



Probabilidad de
que un correo
electrónico sea
spam

$$= \frac{\text{sum}(\text{spam}=1)}{\text{sum}(\text{spam}=1 \text{ or } \text{spam}=0)}$$

$$P(\text{Spam}) = \frac{4}{6}$$

Email	millions	spam
1	1	1
2	0	0
3	1	0
4	0	1
5	1	1
6	1	1

P(million)

¿Cuál es la probabilidad de que un correo electrónico contenga la palabra "million"?

¿Cuántos emails contienen la palabra "million"?



Probabilidad de
Un correo
electrónico que
contiene la
palabra "million"

$$= \frac{\text{sum}(\text{million}=1)}{\text{sum}(\text{million}=1 \text{ or } \text{million}=0)}$$

$$P(\text{million}) = \frac{4}{6}$$

Email	millions	spam
1	1	1
2	0	0
3	1	0
4	0	1
5	1	1
6	1	1

Tarea

Definiendo
 $f(x)$

¡Ahora vamos a juntarlo todo!

$$\begin{aligned} P(\text{Spam} | \text{million}) &= \frac{P(\text{million} | \text{Spam}) * P(\text{Spam})}{P(\text{million})} \\ &= \frac{3/4 * 4/6}{4/6} = 0.75 \end{aligned}$$

Probabilidad de que este correo electrónico sea spam, dado que tiene la palabra "millions" $\equiv 0.75$

Email	millions	spam
1	1	1
2	0	0
3	1	0
4	0	1
5	1	1
6	1	1





Tarea

Predicción

¡Ahora podemos predecir el spam!

¡Ahora podemos predecir el correo no deseado con una sola palabra! Nuestro modelo predice que si la palabra "millones" está presente, es 75% spam. Como este es un problema de clasificación, redondeamos esto a 1.

Email	millions	spam	predicción
1	1	1	1
2	0	0	0
3	1	0	1
4	0	1	0
5	1	1	1
6	1	1	1





Tarea

Predicción

¿Qué hay de las otras palabras?

Pero este modelo es simplista. ¡Deberíamos usar las otras palabras en el correo electrónico también! Podemos repetir el proceso de calcular la probabilidad condicional para todas las palabras en un correo electrónico.

P(Spam | We found out that there is a total amount of \$4.5 million...)

= P(Spam|we) * P(Spam|found) *
P(Spam|total) * P(Spam|amount) *
P(Spam|million) ...

Email	"millions"	"FBI"	"amount"	...
1	1	1	0	
2	0	0	0	
3	1	0	1	
4	0	0	1	
5	1	0	1	
6	1	1	1	



Tarea

Definiendo
 $f(x)$

¿Qué pasa con varias palabras?

Una vez que tenemos probabilidades condicionales para todas las palabras en un correo electrónico, las **multiplicamos** para obtener la probabilidad de que este correo electrónico sea spam, dadas todas las palabras que contiene.

$P(\text{Spam} \mid \text{We found out that there is a total amount of \$4.5 million...})$

$$= P(\text{Spam} \mid \text{we}) * P(\text{Spam} \mid \text{found}) * \\ P(\text{Spam} \mid \text{total}) * P(\text{Spam} \mid \text{amount}) * \\ P(\text{Spam} \mid \text{million}) \dots$$

Email	"millions"	"FBI"	"amount"	...
1	1	1	0	
2	0	0	0	
3	1	0	1	
4	0	0	1	
5	1	0	1	
6	1	1	1	

NOTA: ¿Qué pasa si una probabilidad es 0? ¡Entonces la probabilidad total sería 0! Para evitar este problema, Naive Bayes usa "**Laplace smoothing**," para garantizar que la probabilidad nunca sea 0. Lee más aquí: https://en.wikipedia.org/wiki/Laplace_smoothing



Ahora que sabe cómo funciona el algoritmo Naive Bayes, examinemos algunas de sus desventajas y suposiciones.



Naive Bayes "ingenuamente" supone que cada palabra es independiente de cualquier otra palabra.

Esto se hereda de nuestra representación del texto como una “**bag-of-words**”. Recuerde, un correo electrónico que diga "no es bueno, es malo" se consideraría igual que "no es malo, es bueno" para el algoritmo Naive Bayes.

Como resultado, puede terminar contando dos palabras altamente correlacionadas, como "dólares" y "millones", lo que elevaría incorrectamente la tasa de error.

¿Cómo podemos cuantificar el rendimiento del algoritmo?



Evaluación del rendimiento del algoritmo



Interpretando nuestros resultados para una sola palabra "millones"



Clasificación



$$P(S|W) = 75\%$$



La probabilidad de que el correo electrónico sea spam dada la presencia de la palabra "millones" es del 75%.

corpus
n=100



Rendimiento

Medidas de
rendimiento

¿Cuál es la exactitud de
nuestro modelo?

Recurrimos a nuestros datos de prueba
para evaluar cómo le fue a nuestro modelo.

Exactitud:

etiquetas predichas correctamente
etiquetas predichas

Email	millones	spam
1	1	1
2	0	0
3	1	0
4	0	1
5	1	1
6	1	1



Rendimiento

Medidas de
rendimiento

¿Cuál es la exactitud de
nuestro modelo?

Exactitud:

$$\frac{\text{\# etiquetas predichas correctamente}}{\text{\# etiquetas predichas}}$$

$$= \frac{4}{6} = 66\%$$

Email	millones	spam	predicción
1	1	1	1
2	0	0	0
3	1	0	1
4	0	1	0
5	1	1	1
6	1	1	1



Recap: falso positivo & falso negativo

Tarea de clasificación de spam: es spam el correo?

Falso positivo:

El email se clasifica como spam cuando no lo es.

Falso negativo:

El email no se clasifica como spam cuando lo es.

	They say you did	They say you didn't
You really did	<i>They are right!</i>	"False Negative"
You really didn't	"False Positive"	<i>They are right!</i>

En cualquier modelo de clasificación en el que trabajemos, evaluaremos el rendimiento en FP, FN además de la exactitud.

Rendimiento

Medidas de
rendimiento

¿Cuál es la tasa de falsos negativos?

Tasa de falsos negativos:

$$\frac{\text{\# predicho incorrectamente como spam}}{\text{\# predicciones totales}}$$

Email	millones	spam	predicción
1	1	1	1
2	0	0	0
3	1	0	1
4	0	1	0
5	1	1	1
6	1	1	1



Rendimiento

Medidas de
rendimiento

Tasa de falsos negativos

Tasa de falsos negativos:

$$\frac{\text{\# predicho incorrectamente como spam}}{\text{\# predicciones totales}}$$

$$= \frac{1}{6} = 16\%$$

Email	millones	spam	predicción
1	1	1	1
2	0	0	0
3	1	0	1
4	0	1	0
5	1	1	1
6	1	1	1

Rendimiento

Medidas de
rendimiento

¿Cuál es la tasa de falsos positivos?

Tasa de falsos positivos:

$$\frac{\# \text{ predicho incorrectamente como no spam}}{\# \text{ predicciones totales}}$$

Email	millones	spam	predicción
1	1	1	1
2	0	0	0
3	1	0	1
4	0	1	0
5	1	1	1
6	1	1	1



Rendimiento

Medidas de
rendimiento

Tasa de falsos positivos

Tasa de falsos positivos:

$$\frac{\text{\# predicho incorrectamente como no spam}}{\text{\# predicciones totales}}$$

$$= \frac{1}{6} = 16\%$$

Email	millones	spam	predicción
1	1	1	1
2	0	0	0
3	1	0	1
4	0	1	0
5	1	1	1
6	1	1	1

Fin de la teoría



Acabamos de automatizar la detección de spam

En el próximo módulo, veremos otro algoritmo de NLP.



Felicidades! ¡Terminaste el módulo!

Obtén más información sobre el machine learning de Delta para una buena misión aquí.