UNIVERSIDAD TECNOLÓGICA NACIONAL

FACULTAD REGIONAL CÓRDOBA

Diplomatura en Data Science Aplicada

Agenda

- Repaso normalización
- Segmentación de frase (Normalización)
- Distancia mínima
- Modelo de lenguaje

Info: https://eventos.python.org.ar/events/pyconar2020/

Revisión Clase 1

Tokenización, Normalización

ReGex - Tokenización - Normalización

Expresiones regulares

- Reglas duras, que poseen un fuerte potencial para extraer información.
- Presentan flexibilidad para atacar muchos problemas como clasificación, detección de patrones, etc.
- Si son pensadas como un primer modelo, tener en cuenta métricas.
- Son complejas muchas veces de escribir, y pueden quedar desactualizadas.
- Siempre ve va detrás de la zanahoria.

```
[^ ]*?@[^ ]*?\.[^ ]*
```

Tokenización

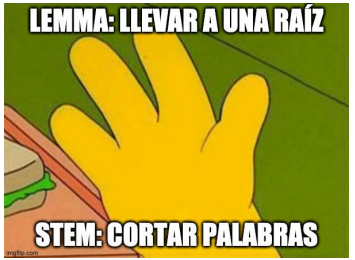
- Enfrentamos el desafío de “contar” palabras (tokens).
- Definimos Tipos (V) y Tokens (N).
- Necesitamos tener en claro cómo resolver problemas más comunes, como OOV, palabras compuestas.
- Evaluar pros y cons de contar con un corpus muy grande.

Natural Language Processing

['Natural', 'Language', 'Processing']

Normalización

- Reducir la complejidad dentro de nuestro corpus a expresiones equivalentes.
- Nos basamos en lematización y stemming para reducir complejidad de los tokens.
- Otras formas, es aplicar expansión asimétrica, equivalencia de clases, minúsculas, etc.



Procesamiento de Texto

Segmentación de frases

Segmentación

El proceso de normalización está compuesto:

- Segmentación o tekenización de nuestro corpus.
- Normalizar palabras o tokens.
- Segmentar las oraciones del corpus.

Encontrar los límites de una frase, oración es uno de los problemas mejores resueltos dentro de NLP. Y está implementado en muchas librerías.

Segmentación - Algunas nociones

Dependiendo los lenguajes (inglés, español) enfrentamos el desafío de cómo definir el final de una frase u oración.

- El punto (**.**), puede presentar varias ambigüedades.
- Signos como: !, ?.

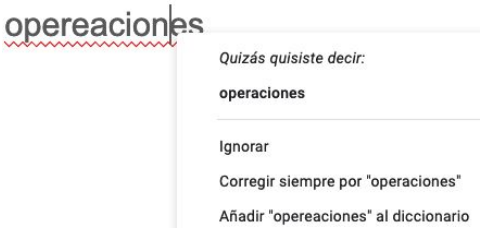
Existen muchas formas de implementarlo desde cero. Por ejemplo:

- Regex
- Modelos de machine learning
  - SVM
  - Árboles de decisión

Distancia de edición Mínima

Distancia mínima de edición - Intuición

Buscamos identificar, de qué tan parecido son dos strings o palabras



Los procesadores de texto están aplicando constantemente este concepto (machine translation).

Cuando activamos los CC en youtube o en una reunión de meet (speech recognition)

*“Distancia de edición mínima (minimum edit distance) es el número mínimo de operaciones de edición que necesitamos realizar, para transformar una palabra en otra **diferente**.”*

Distancia mínima de edición

Encontrar la mínima distancia entre dos strings

Operaciones de edición:

- Inserción
- Borrado
- Sustitución

Transformemos:

- operación → ejecución
- christian → cristhian

O P E R A C I Ó N
| | | | | | | | | |
| | | | | | | | | |
E J E C U C I Ó N

Costo_1 = 4

Costo_2 = 2

Utilizamos la distancia de Levenshtein, la cual asigna diferente pesos a las operaciones. Las sustituciones valen 2

Distancia mínima de edición - Definición formal

Dan Jurafsky

Defining Min Edit Distance (Levenshtein)

- Initialization

  $D(i,0) = i$

  $D(0,j) = j$

- Recurrence Relation:

  For each $i = 1 \ldots M$

  For each $j = 1 \ldots N$

  $$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } X(i) \neq Y(j) \\ 0; & \text{if } X(i) = Y(j) \end{cases} \end{cases}$$

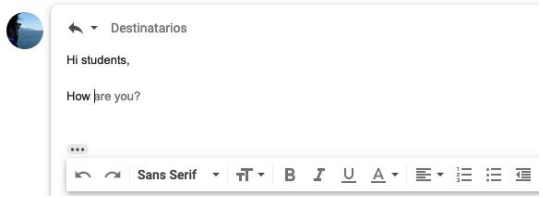
- Termination:

  $D(N,M)$ is distance

Language Modeling

Language modeling - Intuiciones

“Es una herramienta estadística que analiza patrones del lenguaje humano para predecir palabras.”



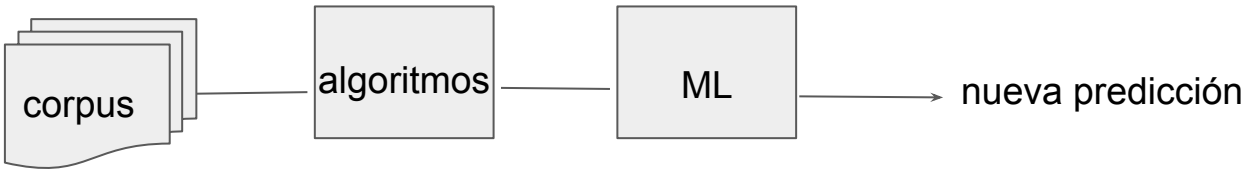
Desafíos para el modelo de lenguaje
Un lenguaje de programación, tiene un conjunto de palabras, reglas, definidas de forma taxativa.

Language modeling es más complejo debido a:

- Ambigüedad propia del lenguaje humano. Una misma cosa puede tener muchos significados.
- La evolución diaria del lenguaje.

Language modeling - Modelado

Asignar una probabilidad (determinada previamente) a una palabra en base a un texto previo.



Existen distintos tipos de modelos

- Modelos Probabilísticos
  - Ngram
  - Exponencial
- Modelos Neuronales
  - LSTM

Language modeling - Modelado

Dan Jurafsky

Probabilistic Language Modeling

- Goal: compute the probability of a sentence or sequence of words:

$$P(W) = P(w_1,w_2,w_3,w_4,w_5 \ldots w_n)$$

- Related task: probability of an upcoming word:

$$P(w_5|w_1,w_2,w_3,w_4)$$

- A model that computes either of these:

$P(W)$ or $P(w_n|w_1,w_2 \ldots w_{n-1})$ is called a **language model**.

Probabilidad Condicional - Recap

“Es la probabilidad asociada a eventos definidos en una subpoblación. Es decir la probabilidad de que se presente un evento, dado la ocurrencia a priori de otro.”

Visto en fórmula:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \quad ; P(B) > 0$$

Siendo:

A, evento que tiene **a** eventos favorables
B, evento que tiene **b** eventos favorables
A∩ B, evento que tiene **c** eventos favorables

	Univ (B)	No univ (D)	Total
Gerencial (A)	25	5	30
No gerencial (C)	75	195	270
Total	100	200	300

Ejemplo:

P(A) = Gerencial = 30/300 = 0,1
P(B) = Universitario = 100/300 = 0,33
A∩ B = Dado que es univ tenga puesto gere
= 25/100 = 0,25
P(A|B) = 25/300 /100/300 = 0,25

Language modeling - Ngram model

“Un modelo basado en N-gram, predice la probabilidad de un N-gram dado dentro de la secuencia de palabras (oración/sentencia).”

Calculemos la probabilidad de los Ngrams

Chain rule probability:

$P(x1, x2, ..., xn) = P(x1)P(x2|x1)P(x3|x1,x2...P(xn|x1,...xn-1)$

$P(\text{can you please come here}) = P(\text{can}) * P(\text{you}|\text{can}) * P(\text{please}|\text{can,you}) * P(\text{come}|\text{can, you, please}) * P(\text{here}|\text{ can,you,please,come})$

Language modeling - Ngram model

Dan Jurafsky

Markov Assumption

$$P(w_1 w_2 \ldots w_n) \approx \prod_i P(w_i \mid w_{i-k} \ldots w_{i-1})$$

- In other words, we approximate each component in the product

$$P(w_i \mid w_1 w_2 \ldots w_{i-1}) \approx P(w_i \mid w_{i-k} \ldots w_{i-1})$$

Language modeling - Ngram model

Calculemos la probabilidad de los Ngrams

Markov Assumption: asumimos que los estados futuros dependen de los estados presentes del modelo.

P(here| can you please come) ~= P(here| come) (uni-grama)

P(here| can you please come) ~= P(here| please come) (bi-grama)

Text Classification

Text Classification - Intuiciones

“Proceso por el cual asigno una etiqueta o categoría a un texto dependendiendo el contenido.”

Ejemplos:

- Detección de spam: **ham** or spam?
- Atribución de autoría sobre obras o artículos
- Clasificación de artículos:
  - político, deportes, policial, etc.
- **Análisis de Sentimiento**:
  - algo es positivo o negativo?
  - También puede asumir otros estados, como neutral, etc
  - Nivel de intensidad emocional.

Text Classification - Definición formal

Dan Jurafsky

Text Classification: definition

- *Input*:
  - a document d
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$
- *Output*: a predicted class $c \in C$

Text Classification - Tipos de clasificadores

Dan Jurafsky

Classification Methods: Supervised Machine Learning

- *Input:*
  - a document d
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$
  - A training set of m hand-labeled documents $(d_1, c_1), \ldots, (d_m, c_m)$
- *Output:*
  - a learned classifier $\gamma: d \rightarrow c$

Text Classification - Tipos de clasificadores

Dan Jurafsky

Classification Methods: Supervised Machine Learning

- Any kind of classifier
  - Naïve Bayes
  - Logistic regression
  - Support-vector machines
  - k-Nearest Neighbors

Anexo

Trabajando con Git

Qué es git?

Es un software de control de versiones distribuido diseñado por Linus Torvals, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de archivos.

Qué es un software de control de versiones?

Es un sistema que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo, de modo que puedas recuperar versiones específicas más adelante.

Es un lugar de almacenamiento (local o remoto) del cual pueden ser recuperados e instalados los paquetes de software.
Ej de repositorio remoto: https://github.com/nestornav/nlp-intro

¿Para qué nos sirve git?

- revertir archivos a un estado anterior.
- revertir el proyecto entero a un estado anterior .
- comparar cambios y quién los hizo.
- trabajar en equipo sobre un mismo proyecto de forma cómoda y eficiente

Estados principales de los archivos

Modificado (*modified*): has modificado el archivo pero todavía no lo has confirmado a tu base de datos.

preparado (*staged*): has marcado un archivo modificado en su versión actual para que vaya en tu próxima confirmación.

Confirmado (*committed*): los cambios, almacenados en tu base de datos local.


git
FALSE
GitHub