CUNEF

❖ EXAMEN GESTIÓN DE LA INFORMACIÓN NO ESTRUCTURADA (TEXT MINING)

1ª Parte: Teoría

- ***Pre-procesamiento de textos. Enumere las tareas que lo componen. ¿En qué se diferencian stemming y lematización? Ponga un ejemplo.***

1. Constitución del Corpus

2. Detección del idioma

3. Segmentación de frases

4. Tokenización: dividir cada una de las frases en tokens (palabras)

5. Normalización:

   5.1 **Stemming**: consiste en extraer la raíz de las palabras y trabajar con ellas en lugar de con la palabra completa en la representación del modelo de documentos.

   5.2 Formas canónicas (**Lematización**): consiste en transformar las palabras en su forma canónica, es decir, hacer infinitivo a los verbos y, nombres y adjetivos establecerlos en el masculino singular.

   Para realizar una correcta lematización hay que hacer no sólo un análisis morfólogico de la palabra, sino también un análisis sintáctico.

Ejemplo: Imaginemos que en un texto encontramos la siguiente frase: “Suelo ir de vacaciones al sur”

6. Análisis morfológico (gramatical), para determinar que si una palabra es un sustantivo, adjetivo, artículo, adjetivo, número, nombre propio… etc

7. Análisis sintáctico, con el fin de poner la frase en contexto y evitar la desambiguación.

8. Shallow parsing, consiste en llevar a cabo un análisis superficial del texto de los documentos.

CUNEF

2. ¿Cuáles son los principales mecanismos de evaluación de un sistema de clasificación atendida? ¿Es suficiente con dar un solo parámetro para ver la bondad de un sistema de clasificación?

La Categorización o clasificación de textos es una de las necesidades fundamentales a las que responde la minería de textos y consiste en asignar cada texto a un conjunto de categorías definidos de antemano.

Un sistema de ***clasificación atendida***, se diferencia de los clasificadores "hard" (no atendida) en que nos va a ofrecer para cada pareja de documento/categoría un valor entre 0 y 1, siendo el valor 1 el de máxima probabilidad de que el documento pertenezca a dicha categoría. Estos son conocidos como clasificadores **"soft".**

A la categorización supervisada o atendida se le denomina "clasificación" mientras que a la no supervisada se le denomina "clustering"

Existen múltiples algoritmos de clasificación supervisada. En principio repasaremos los probabilísticos y lineales más frecuentes: – Árboles de decisión – Clasificación Bayesiana – K-nearest neighbors – Vector Machine – Modelos de máxima entropía.

3. Un Banco de Consumo de tamaño medio desea agilizar su proceso de tramitación de quejas y reclamaciones de los clientes. Actualmente el primer paso es una clasificación manual realizada por empleados del banco. Este proceso es lento y tiene un coste alto. ¿Qué sistema propondría usted para mecanizar este procedimiento? Indique si emplearía un software comercial, software de libre distribución o realizaría un programa a medida. Indique también el algoritmo recomendado.

Se denomina sistema supervisado a aquel proceso que requiere un trabajo manual previo, lo que puede ser lento y costoso.

En este caso, emplearía un software de libre distribución como SOLR

Solr (pronunciado "solar") es un proyecto open source basado en Lucene (software libre de Apache) y escrito en Java. Como servidor de búsqueda, **Apache Solr** es una de las herramientas más populares para **integrar motores de búsqueda verticales**. Las ventajas de Solr comprenden una amplia gama de funciones (incluyendo, por ejemplo, el facetado de los resultados de búsqueda) y la indexación acelerada. Se ejecuta en contenedores-servidor como Tomcat (Apache).

Como algoritmo clasificador más eficiente en este caso usaría un ***NAIVE BAYES CLASIFICATOR*** debido a que producen resultados comparables con los obtenidos por otros m´etodos m´as sofisticados y son relativamente sencillos de implementar.

El clasificador Naive Bayes Simple considera la probabilidad de aparici´on de cada t´ermino dada la clase de forma binaria, es decir el t´ermino aparece o no y entonces su probabilidad condicional dada la clase es o no considerada. En este sentido, el clasificador Naive Bayes Multinomial suele mejorar el desempeño pues considera el número de apariciones del término para evaluar la contribución de la probabilidad condicional dada la clase con lo que el modelado de cada documento se ajusta mejor a la clase a la que pertenece

4.Enumere los principales paquetes open-source de PLN. Describa brevemente la arquitectura de dichos paquetes.

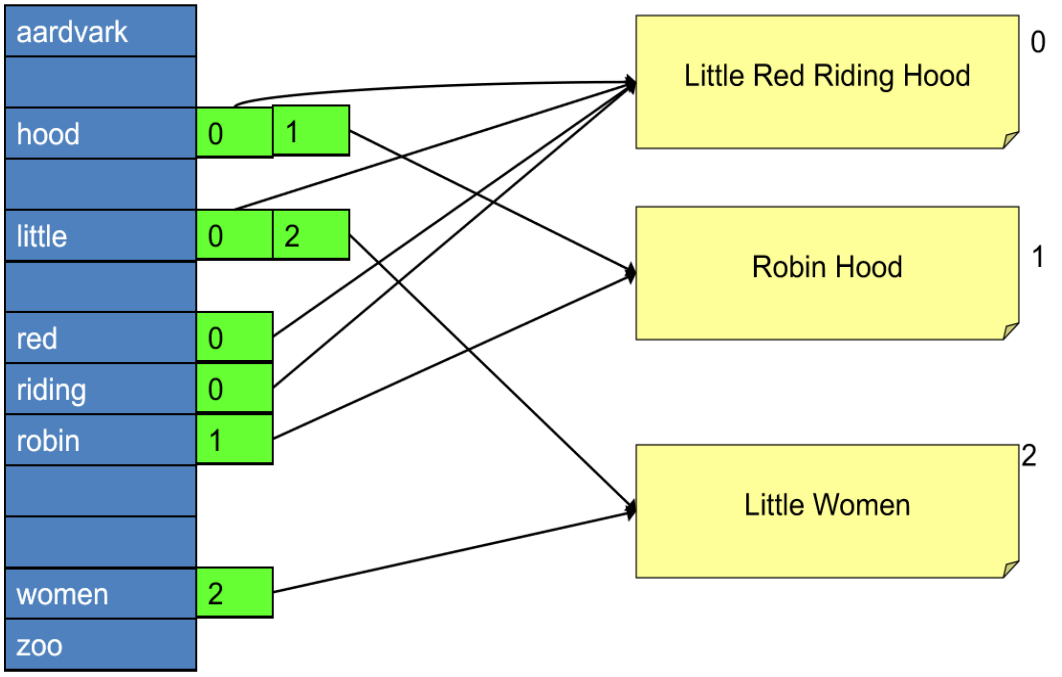
La arquitectura UIMA (Unstructured Information Management Architecture) - Arquitectura y gestión de Información no estructurada) es especialmente pensada para combinar los distintos componentes de la fase de análisis de textos y facilitar su disponibilidad para diversas aplicaciones en la fase de entrega. El componente que contiene lógica se llama anotador (annotator). Cada anotador realiza una tarea específica de extracción de información de un documento y genera como resultado, anotadores que se añaden a una estructura de datos llamada CAS (Common Analysis Structure).

- ***LUCENE.***

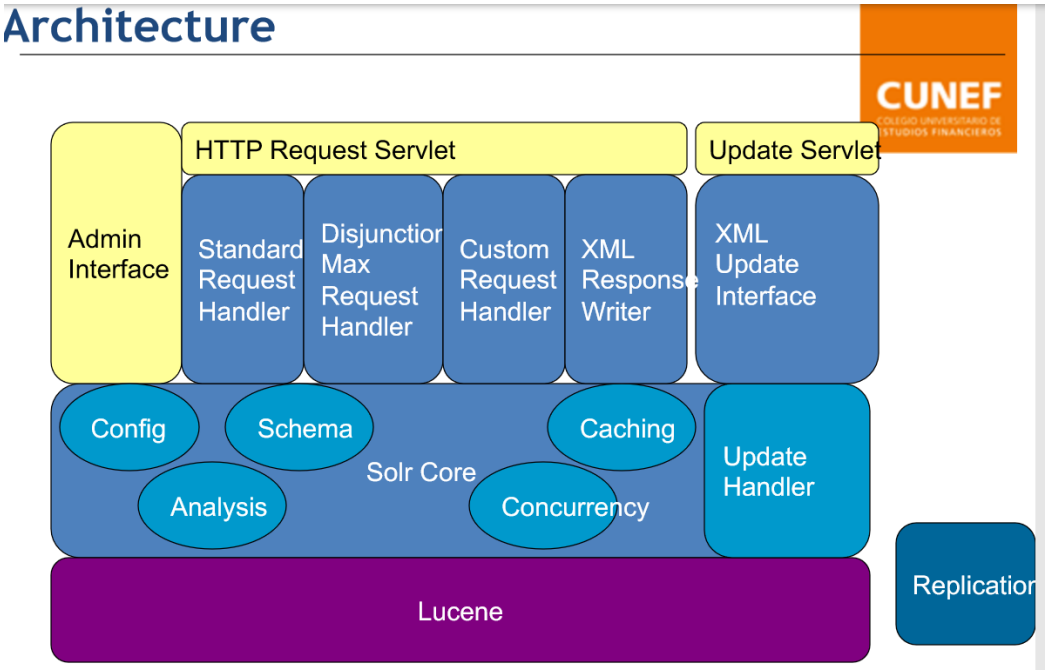
Lucene is a full-text search library •Add documents to an index via IndexWriter – A document is a a collection of fields – No config files, dynamic field typing – Flexible text analysis – tokenizers

CUNEF

Inverted Index



- ***SOLR***: Se basa en LUCENE, XML/HTTP Interfaces



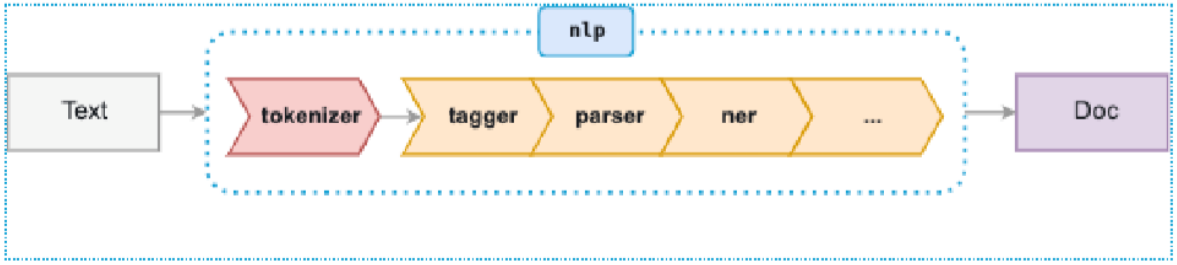
- ***ELASTICSEARCH***

Flexible y poderoso open source, distribuye analytics en motores de búsquedas REAL TIME para la nube.

- spaCY

spaCy es una librería ***open-source*** para realizar Procesamiento de Lenguaje Natural en Python • spaCy se ha diseñado de forma específica buscando productividad • Incorpora modelos que le permiten trabajar en múltiples idiomas.

La forma de trabajar de spaCy es construir una pipeline con el modelo adecuado al idioma de trabajo, y ejecutarla sobre un texto para conseguir un objeto de tipo "Doc"



5. ¿Cuáles son los algoritmos más frecuentes que se utilizan para la extracción de tópicos? ¿Qué librerías Python son las más utilizadas que implementen dichos algoritmos?

En cuanto al modelado de tópicos, los algoritmos más utilizados para la extracción de tópicos son Latent Dirichlet Allocation (**LDA**), Probabilistic Latent Semantic Analysis (**PLSA**) y Correlated Topic Model (**CTM**)

- ***Latent DirichletAllocation***

Se trata de un modelo generativo (genera tópicos o categorías) que permite explicar un conjunto de observaciones, agrupándolos en clusters. La razón para que un documento esté en un cluster es que tenga datos similares al conjunto de documentos que hay en él.

Para LDA, asumimos que cada documento es una mezcla de tópicos, reflejados en palabras, con cierta probabilidad (frecuencias). La distribución de la mezcla de tópicos sigue una distribución de Dirichlet.

Entre las librerías más destacadas para implementar dichos algoritmos encontramos entre otras:

- nltk, re, pprint, spacy, numpy
- RegexpTokenizer
- PlaintextCorpusReader
- get_stop_words
- PorterStemmer
- from sklearn.decomposition import LatentDirichletAllocation
- from sklearn.feature_extraction.text import CountVectorizer
- from nltk.stem import WordNetLemmatizer
- import textacy
- corpora, models
- gensim
- pyLDAvis.gensim