

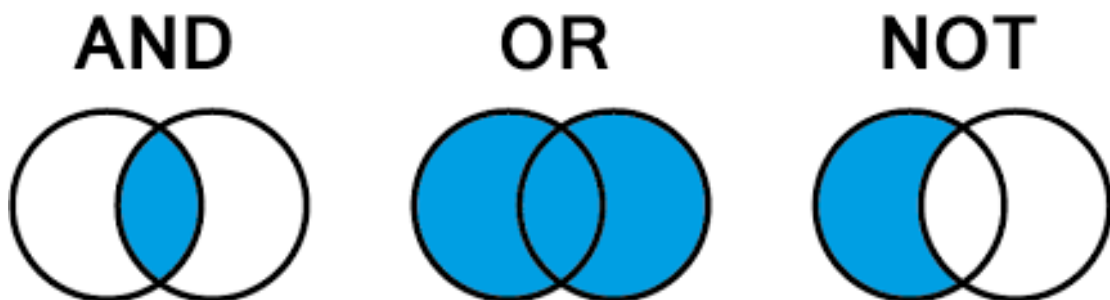


INSTITUTO TECNOLÓGICO BELTRÁN
Centro de Tecnología e Innovación

Informe

Trabajo Práctico N.º 7

Modelo De Claves Booleanas



Nombre: Coral Tolazzi
Tema: Recuperación de la Información
Profesora: Yanina Ximena Scudero
Cuatrimestre y Año: 1 Cuatrimestre del 2025

Instituto tecnológico Beltrán
Procesamiento del Lenguaje Natural

EJERCICIO:

Crear un programa en Python que permita al usuario realizar búsquedas booleanas (AND, OR, NOT) en un conjunto de documentos utilizando NLTK.

Parte 1: Civilizaciones Antiguas

Se tienen 5 documentos con información sobre civilizaciones antiguas:

"doc1": "Los egipcios construyeron las pirámides y desarrollaron una escritura jeroglífica.",




"doc2": "La civilización romana fue una de las más influyentes en la historia occidental.",

"doc3": "Los mayas eran expertos astrónomos y tenían un avanzado sistema de escritura.",

"doc4": "La antigua Grecia sentó las bases de la democracia y la filosofía moderna.",

"doc5": "Los sumerios inventaron la escritura cuneiforme y fundaron las primeras ciudades."

Ejemplos de consulta:

- Ingrese una consulta booleana (o 'salir' para terminar): egipcios AND pirámides
 Documentos encontrados: {'doc1'}
- Ingrese una consulta booleana (o 'salir' para terminar): escritura OR astrónomos
 Documentos encontrados: {'doc1', 'doc3', 'doc5'}
- Ingrese una consulta booleana (o 'salir' para terminar): romana NOT griegos
 Documentos encontrados: {'doc2'}

Parte 2: Inteligencia Artificial y Aprendizaje Automático

Se tienen 5 documentos con información sobre inteligencia artificial:

"doc1": "La inteligencia artificial está revolucionando la tecnología.",

"doc2": "El aprendizaje automático es clave en la inteligencia artificial.",

"doc3": "Procesamiento del lenguaje natural y redes neuronales.",

"doc4": "Las redes neuronales son fundamentales en deep learning.",


"doc5": "El futuro de la IA está en el aprendizaje profundo."

Ejemplos de consulta:


Ingrese una consulta booleana (o 'salir' para terminar): inteligencia AND artificial

 Documentos encontrados: {'doc1', 'doc2'}

Ingrese una consulta booleana (o 'salir' para terminar): redes OR aprendizaje

 Documentos encontrados: {'doc2', 'doc3', 'doc4', 'doc5'}

Ingrese una consulta booleana (o 'salir' para terminar): inteligencia NOT automático

 Documentos encontrados: {'doc1'}

Requisitos:

- Implementa una función que tokenice y limpie los documentos.
- Crea un índice invertido (asocia cada palabra clave a los documentos en los que aparece).

Explicacion DelCodigo ModelosDeClasesBooleanas:

El programa permite al usuario realizar búsquedas booleanas (AND, OR, NOT) sobre documentos de texto (de Historia o de Inteligencia Artificial), utilizando:

- NLTK para preprocesar texto (tokenizar y eliminar palabras vacías).
- Whoosh para crear un índice invertido y realizar búsquedas eficientes.

Estructura del código

1. Importaciones

```
import nltk
from nltk.tokenize import word_tokenize
from whoosh.index import create_in
from whoosh.fields import Schema, TEXT
from whoosh.qparser import QueryParser
```

- nltk: se usa para preprocesar texto.
- whoosh: biblioteca que permite crear un motor de búsqueda local.

2. Datos de ejemplo

```
documentosDeIa = {...}
documentosDeHistoria = {...}
```

Dos diccionarios con textos. Uno sobre IA y otro sobre civilizaciones antiguas, con identificadores doc1, doc2, etc.

3. Stopwords en español

```
stop_words = set(nltk.corpus.stopwords.words('spanish'))
```

Lista de palabras vacías (como “de”, “la”, “y”) que se eliminan durante el preprocesamiento.

4. Preprocesamiento

```
def preprocess(text):
    tokens = word_tokenize(text.lower())
    return [word for word in tokens if word.isalnum() and word not in stop_words]
```

- Convierte el texto a minúsculas.
- Tokeniza el texto (lo divide en palabras).
- Elimina signos de puntuación y palabras vacías.

5. Crear índice Whoosh

```
def crear_indice(documents):
    schema = Schema(title=TEXT(stored=True), content=TEXT(stored=True))
    indice = create_in("indexdir", schema)
```

```
writer = indice.writer()
...
return indice
```

- Define el esquema (estructura del documento): título y contenido.
- Crea un índice en una carpeta llamada "indexdir".
- Agrega cada documento preprocesado al índice.

6. Buscar en el índice

```
def buscar(indice):
    with indice.searcher() as searcher:
        parser = QueryParser("content", indice.schema)
        ...
```

- Permite al usuario ingresar consultas booleanas (AND, OR, NOT).
- Usa el parser de Whoosh para interpretar la consulta.
- Muestra los documentos que coinciden.
- Si no se encuentra nada, imprime un mensaje adecuado.

7. Bloque principal del programa

```
if __name__ == "__main__":
    ...
```

- Pide al usuario elegir entre documentos de IA o Historia.
- Crea el índice con los documentos seleccionados.
- Llama a la función buscar() para comenzar las consultas.

Consultas Del Codigo ModelosDeClasesBooleanas:

Documentos de Civilizaciones Antiguas:

```
Ingrese 1 si quiere consultar utilizando los documentos sobre IA o 2 si quiere utilizar los documentos sobre Historia: 2
Búsqueda booleana en documentos sobre Historia
Ingrese una consulta booleana (o 'salir' para terminar): egipcios AND p
irámides
Documentos encontrados: {'doc1'}
Ingrese una consulta booleana (o 'salir' para terminar): escritura OR a
strónomos
Documentos encontrados: {'doc1', 'doc3', 'doc5'}
Ingrese una consulta booleana (o 'salir' para terminar): romana NOT griegos
Documentos encontrados: {'doc2'}
Ingrese una consulta booleana (o 'salir' para terminar): salir
```

Documentos de IA y aprendizaje automático:

```
Ingrese 1 si quiere consultar utilizando los documentos sobre IA o 2 si quiere utilizar los documentos sobre Historia: 1
Búsqueda booleana en documentos sobre IA y aprendizaje automático
Ingrese una consulta booleana (o 'salir' para terminar): inteligencia AND artificial
Documentos encontrados: {'doc1', 'doc2'}
Ingrese una consulta booleana (o 'salir' para terminar): redes OR aprendizaje
Documentos encontrados: {'doc5', 'doc4', 'doc3', 'doc2'}
Ingrese una consulta booleana (o 'salir' para terminar): inteligencia NOT automático
Documentos encontrados: {'doc1'}
Ingrese una consulta booleana (o 'salir' para terminar): salir
```