Contar palabras

Viviana Marcela García Valderrama- 506222720
Fundación Universitaria Konrad Lorenz

I. Introducción

En este informe, nos enfocaremos en una técnica fundamental para extraer información valiosa de conjuntos de documentos: el análisis de palabras. Utilizando Python, hemos llevado a cabo un análisis detallado de un conjunto de documentos con el objetivo de identificar las palabras más frecuentes.

Además de analizar las palabras más repetidas, tambien se ha incorporado una funcionalidad que permite a los usuarios buscar palabras específicas dentro de estos documentos. Esto brinda a los usuarios la capacidad de interactuar con el análisis y obtener información más específica sobre la presencia de palabras clave en los textos que hemos estudiado.

A continuación, se presentará el proceso de análisis de palabras y la funcionalidad de búsqueda de palabras en detalle, destacando su importancia en la extracción de información relevante de grandes conjuntos de documentos.

II. Origen de los Documentos

Los documentos utilizados en este análisis son una muestra de textos relacionados con la programación y el desarrollo de software. Estos documentos representan una variedad de temas en el campo de la informática y se han recopilado con el propósito de realizar un análisis de palabras clave.

A continuación, se presenta la lista de documentos utilizados en el análisis:

III. Lista de documentos utilizados en el análisis

```python
import time
import psutil


# Lista de documentos
my_documents = [
    "La programación en Python es clave para el trabajo con datos",
    "Los programadores en Java tienen un alto interés en pasar a Python",
    "La optimización de algoritmos es fundamental en el desarrollo de software",
    "Las bases de datos relacionales son esenciales para muchas aplicaciones",
    "El paradigma de programación funcional gana popularidad",
    "La seguridad informática es un tema crucial en el desarrollo de aplicaciones web",
    "Los lenguajes de programación modernos ofrecen abstracciones poderosas",
    "La inteligencia artificial está transformando diversas industrias",
    "El aprendizaje automático es una rama clave de la ciencia de datos",
    "Las interfaces de usuario intuitivas mejoran la experiencia del usuario",
    "La calidad del código es esencial para mantener un proyecto exitoso",
    "La agilidad en el desarrollo de software permite adaptarse a cambios rápidamente",
    "Las pruebas automatizadas son cruciales para garantizar la estabilidad del software",
    "La modularización del código facilita la colaboración en equipos de programadores",
    "El control de versiones es necesario para rastrear cambios en el código",
    "La documentación clara es fundamental para que otros entiendan el código",
    "La programación orientada a objetos promueve la reutilización de código",
    "La resolución de problemas es una habilidad esencial en la programación",
    "La optimización prematura puede llevar a código complicado y difícil de mantener",
    "El diseño de interfaces de usuario atractivas mejora la usabilidad de las aplicaciones",
    "El código limpio es esencial para facilitar el mantenimiento",
```

Figura 1. Parte 1

```
    "Los patrones de diseño son soluciones probadas para problemas comunes",
    "Las pruebas unitarias garantizan el correcto funcionamiento de las partes del código",
    "El desarrollo ágil prioriza la entrega continua de valor al cliente",
    "Los comentarios en el código deben ser claros y útiles",
    "La recursividad es una técnica poderosa en la programación",
    "Las bibliotecas de código abierto aceleran el desarrollo de software",
    "La virtualización permite una mejor utilización de los recursos de hardware",
    "La seguridad en la programación web es fundamental para prevenir ataques",
    "Los principios SOLID son fundamentales para el diseño de software robusto",
    "La arquitectura de microservicios permite escalar componentes individualmente",
    "La refactorización mejora la calidad del código sin cambiar su comportamiento",
    "Los sistemas distribuidos presentan desafíos en la sincronización de datos",
    "El enfoque DevOps une el desarrollo y las operaciones para una entrega eficiente",
    "Las bases de datos NoSQL son útiles para manejar datos no estructurados",
    "La agilidad en el desarrollo permite adaptarse a cambios del mercado",
    "Las buenas prácticas en el control de versiones facilitan la colaboración",
    "La programación concurrente mejora la eficiencia en sistemas multiusuario",
    "Los marcos de trabajo MVC separan la lógica de la interfaz de usuario",
    "La interacción entre aplicaciones se logra a través de APIs",
    "El machine learning permite a las máquinas aprender de los datos",
    "La analítica de datos ayuda a tomar decisiones basadas en información",
    "El diseño responsivo garantiza una experiencia consistente en diferentes dispositivos",
    "Las pruebas de carga verifican el rendimiento de las aplicaciones",
    "El enfoque centrado en el usuario mejora la usabilidad de las aplicaciones",
    "La programación reactiva es útil para manejar flujos de datos asincrónicos",
    "Los contenedores facilitan la implementación y el despliegue de aplicaciones",
    "La gestión de dependencias es esencial para administrar las bibliotecas externas",
```

Figura 2. parte 2

```
    "La integración continua automatiza la verificación de cambios en el código",
    "El aprendizaje profundo es una rama avanzada del machine learning",
    "La depuración es una habilidad crucial para encontrar y corregir errores",
    "La criptografía protege la información sensible en aplicaciones",
    "El desarrollo full-stack abarca tanto el frontend como el backend",
    "Las pruebas de seguridad ayudan a identificar vulnerabilidades en el software",
    "La agilidad cultural es clave para adoptar prácticas ágiles de manera efectiva",
    "La infraestructura como código permite automatizar la gestión de servidores",
    "Los patrones arquitectónicos guían la estructura general de una aplicación",
    "El análisis predictivo utiliza datos históricos para predecir tendencias",
    "Las interfaces API REST son ampliamente utilizadas para comunicarse con aplicaciones",
    "El rendimiento de las aplicaciones es esencial para brindar una buena experiencia",
    "La virtualización de servidores reduce costos y facilita la administración",
    "La ingeniería de software implica la aplicación de métodos sistemáticos",
    "El código autodocumentado es claro y fácil de entender para otros programadores",
    "La integración de sistemas conecta diferentes aplicaciones para trabajar juntas",
    "Las metodologías ágiles promueven la adaptación y la colaboración continua",
    "El monitoreo de aplicaciones permite identificar y resolver problemas en tiempo real",
    "El análisis de datos masivos (big data) abre oportunidades para obtener insights",
    "El diseño de interfaces de usuario es crucial para la experiencia del usuario",
    "La seguridad en el desarrollo es un proceso constante de mitigación de riesgos"
]
```

Figura 3. parte 3

IV. Código en Python

A continuación, se presenta el código Python utilizado para el análisis de palabras.

```python
# Funci n para contar palabras repetidas
def contar_palabras_repetidas(documentos):
    contador = {}  # O(1)
    for documento in documentos:  # O(n)
        palabras = documento.lower().split()  # O(m)
        for palabra in palabras:  # O(m)
            if palabra in contador:  # O(1) en promedio, O(m) en el peor caso
                contador[palabra] += 1  # O(1)
            else:
                contador[palabra] = 1  # O(1)

    # Ordenar el contador por frecuencia de mayor a menor
    palabras_ordenadas = dict(sorted(contador.items(), key=lambda item: item[1], reverse=True))  # O(m*log(m))
    return palabras_ordenadas

# Funci n para buscar documentos que contienen una palabra espec fica
def buscar_documentos_por_palabra(documentos, palabra):
    documentos_contienen_palabra = []
    for i, documento in enumerate(documentos):  # O(n)
```

```
        if palabra in documento.lower():  # O(m)
            documentos_contienen_palabra.append(i)
                # O(1)
    return documentos_contienen_palabra

# Llamada a la funci n para contar palabras
    repetidas
palabras_contadas =
    contar_palabras_repetidas(my_documents)

# Solicitar la palabra al cliente por teclado
palabra_buscada = input("Ingresa la palabra que
    deseas buscar: ")

# Tiempo de ejecuci n
start_time = time.time()  # O(1)
memoria_inicio = psutil.virtual_memory().used  #
    O(1)

# Imprimir palabras m s repetidas de mayor a menor
print("Palabras m s repetidas de mayor a menor:")
for palabra, frecuencia in
    palabras_contadas.items():  # O(m)
    print(f"{palabra}: {frecuencia}")

# Buscar documentos que contienen la palabra
    ingresada por el cliente
documentos_con_palabra =
    buscar_documentos_por_palabra(my_documents,
    palabra_buscada)
print("----------------------------")
print(f"Documentos que contienen la palabra
    '{palabra_buscada}': {documentos_con_palabra}")

# Tiempo y memoria empleados
end_time = time.time()  # O(1)
memoria_final = psutil.virtual_memory().used  # O(1)

print("------------------------------")

print(f"El tiempo que se gast en ejecutarse es:
    {end_time - start_time} segundos")  # O(1)
print(f"La memoria que se utiliz fue:
    {memoria_final - memoria_inicio} bytes")  # O(1)

print("------------------------------")

# La complejidad total del c digo es O(n * m ).
```

V. Análisis de Palabras

Acontinuación, se presentan los resultados del análisis de palabras.

V-A. *Palabras más repetidas*

Aquí se muestran las 10 palabras más repetidas de mayor a menor:

- de: 59 veces
- la: 58 veces
- el: 40 veces
- en: 24 veces
- para: 23 veces
- es: 20 veces
- las: 19 veces
- código: 14 veces
- Aplicaciones: 12 veces
- datos: 11 veces

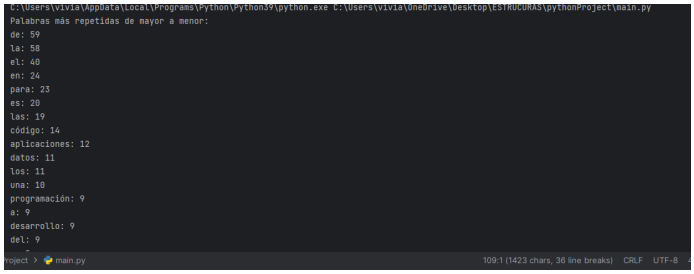

Figura 4. Palabras mas repetidas.

V-B. *Tiempo de ejecución y uso de memoria*

El tiempo de ejecución del análisis fue de 0.010082006454467773 segundos y se utilizó un total de 1789952 bytes de memoria.

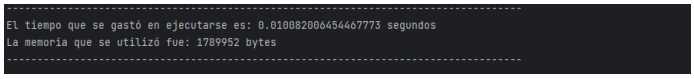

Figura 5. Tiempo y memoria usados en la ejecución

VI. Búsqueda de Palabras

En este análisis,los usuarios pueden ingresar una palabra de su elección, y el sistema buscará esa palabra en todos los documentos y mostrará en cuáles documentos aparece. A continuación, se muestra un fragmento del codigo de búsqueda que permite al usuario buscar una palabra en especifico:

```
# Solicitar la palabra al cliente por teclado
palabra_buscada = input("Ingresa la palabra que
    deseas buscar: ")

# Buscar documentos que contienen la palabra
    ingresada por el cliente
documentos_con_palabra =
    buscar_documentos_por_palabra(my_documents,
    palabra_buscada)
print("----------------------------")
print(f"Documentos que contienen la palabra
    '{palabra_buscada}': {documentos_con_palabra}")
```

Este ejemplo muestra cómo los usuarios pueden interactuar con el análisis para buscar palabras específicas y obtener una lista de documentos que contienen esa palabra.

A continuación, se presenta la lista de índices de documentos que contienen la palabra "las" en la ejecución que se hizo para la prueba:

Figura 6. Lista de documentos que contienen la palabra las

VII. Conclusión

En este informe hemos realizado un análisis detallado de un conjunto de documentos relacionados con la programación y el desarrollo de software. Utilizando Python, hemos identificado las palabras más frecuentes en estos documentos, lo que proporciona una visión importante de los temas y términos clave en el campo de la informática.

Además, hemos introducido un funcionalidad que permite a los usuarios buscar palabras específicas en estos documentos. Esto amplía la utilidad del análisis al brindar a los usuarios la capacidad de interactuar directamente con los datos y obtener información más específica sobre las palabras clave que les interesan.

En términos de eficiencia, hemos observado que el análisis de palabras y la búsqueda de palabras se realizan de manera rápida y eficiente en Python, lo que demuestra cómo las herramientas de procesamiento de texto en este lenguaje pueden facilitar la extracción de información valiosa de grandes conjuntos de documentos.