Actividad 4

Clasificador con Python

Objetivo

Resolver de forma original y en grupos el proceso de clasificación de textos implementando una aplicación con interfaz gráfica en Python.

Descripción del trabajo a desarrollar

Tras las pruebas de clasificadores de texto realizadas anteriormente con RapidMiner, se pretende desarrollar una aplicación en Python que implemente un clasificador de texto, que permita diferenciar entre las categorías predefinidas.

Tal y como se ha venido trabajando, idealmente se tendrá:

- Una categoría "**odio**" representada a través de una carpeta de idéntico nombre. Contiene textos (noticias, comentarios en redes sociales) que hacen referencia al tema de los delitos de odio. Se han descargado de periódicos online y/o redes sociales, y se han almacenado en ficheros en formato de texto plano.
- La categoría de "no odio", representada mediante una carpeta "**no_odio**". Contiene (noticias, comentarios en redes sociales) que tratan temas distintos a los de delitos de odio (cultura, economía, deportes, ocio, etc.). Se han descargado de periódicos online y/o redes sociales y se han almacenado en ficheros en formato de texto plano.
- Una carpeta con documentos cuya categoría aún se desconoce ("**unlabeled**"), que serán la entrada de la herramienta, y que se clasificarán según las categorías anteriores.
- Opcionalmente: un fichero de texto con las palabras que formarán la lista de parada (*stop words*).

Se pide:

- Entrenar un clasificador de texto capaz de distinguir entre las categorías anteriores ("odio" y "no odio") a partir de los ficheros de texto plano almacenados en las correspondientes carpetas.
- Permitir **elegir entre al menos tres algoritmos de aprendizaje diferentes**. Idealmente serían los tres mejores resultantes de las pruebas de la actividad anterior (RapidMiner).

- Para ello habrá que procesar el texto de las noticias. Será necesario realizar como mínimo los siguientes pasos: *tokenización*, conversión a minúsculas, lista de parada y *stemming / lematización*.
- Ejecutar el modelo con una reserva de documentos no utilizados en el entrenamiento, para validar su funcionamiento.
- Presentar datos estadísticos de los procesos de entrenamiento y clasificación.
- Permitir guardar diferentes modelos entrenados y resultados de clasificación.
- Documentación del trabajo realizado.

Documentación

El trabajo deberá documentarse convenientemente. La siguiente es una propuesta de guion a seguir:

1. Portada
2. Resumen (1 ó 2 párrafos explicando qué hace el sistema)
3. Índice
4. Introducción
   4.1. Contextualización
   4.2. Descripción del problema
   4.3. Cómo se organiza el resto del documento
5. Estado de la cuestión
6. Solución
   6.1. Objetivos generales y específicos
   6.2. Descripción de la solución propuesta:
      - Qué hace y qué no hace la aplicación
      - Tecnologías / librerías empleadas
      - Esquema gráfico de la arquitectura/organización del proyecto
      - Descripción de los datos de entrada
      - Datos de salida / Resultados
      - Tratamiento del texto (tokenización, stemming, etc.)
      - Proceso de entrenamiento (descripción de los algoritmos utilizados y su implementación)
7. Diseño
   7.1. Fuentes de datos y dataset generado
   7.2. Diagrama UML de clases
8. Metodología de trabajo
   8.1. Plan de trabajo (roles, tiempos, tareas, etc.)
9. Presupuesto
10. Diseño de pruebas y resultado de las mismas

11. Manual de instalación
    11.1. Requeriments.txt
12. Manual de usuario
13. Conclusiones
14. Trabajos futuros
15. Bibliografía

***Nota:** El apartado de conclusiones de un documento como el que nos ocupa no es un punto donde damos nuestra opinión personal sobre "lo mucho que he aprendido" o "lo complicado que ha sido hacer" una tarea concreta. En este apartado se debe analizar todo lo expuesto en los anteriores, y razonar qué se puede concluir de lo expuesto. El rigor exige que la conclusión se base en contenidos previamente expuestos y debidamente documentados. Es decir, que no se pueden establecer conclusiones basadas en hipótesis no verificables o difícilmente comprobables.

Normas de entrega

Seguir los siguientes pasos antes de realizar la entrega:

1. Crear una carpeta con el siguiente formato de nombre: Grupo_X.PC1.Act4, donde la X indica el número del grupo de trabajo.
2. Crear dentro de ella dos subcarpetas: "Doc" y "Código".
3. La carpeta Doc contendrá:
   - Un fichero **PDF** con la memoria/documentación de la actividad: Grupo_X.PC1.Act4**.PDF**
   - Una carpeta "odio": contendrá 50 ficheros de ejemplo de textos de odio usados para entrenar.
   - Una capeta "no_odio"Doc: contendrá 50 ficheros de ejemplo de textos de no odio usados para entrenar.
   - Una carpeta "unlabeled": contendrá 10 ficheros de ejemplo de textos desconocidos.
4. Carpeta "Código": contendrá los ficheros fuente y el *requeriments.txt,* necesarios para poder replicar y ejecutar el proyecto.
5. Comprimir la carpeta en un fichero **ZIP** (no se aceptarán ficheros RAR, 7Z u otro formato que no sea ZIP). El resultado será un fichero con el siguiente nombre:

   Grupo_X.PC1.Act4.**ZIP**

6. Subir el fichero a la actividad **antes de las 23:55h** del día marcado como límite.