Programación para la ciencia de datos - PEC4

En este Notebook encontraréis el ejercicio que supone la cuarta y última actividad de evaluación continuada (PEC) de la asignatura. Esta PEC intenta presetaros un pequeño proyecto en el cual debéis resolver diferentes ejercicios, que engloba muchos de los conceptos cubiertos durante la asignatura.

El objetivo de este ejercicio será desarrollar un **paquete de Python** fuera del entorno de Notebooks, que nos permita resolver el problema dado. Trabajaréis en archivos Python planos `.py`. Este tendrá que incluir el correspondiente código organizado lógicamente (separado por módulos, organizados por funcionalidad,...), la documentación del código (*docstrings*) y tests. Además, tendréis que incluir los correspondientes archivos de documentación de alto nivel (`README`) así como los archivos de licencia y dependencias (`requirements.txt`) comentados en la teoría.

Hacer un `setup.py` es opcional, pero si se hace se valorará positivamente de cara a la nota de la práctica y del curso.

Enunciado:

Nos han encargado analizar el contenido de una base de datos de Twitter para un proyecto de procesamiento del lenguaje natural (NLP) relacionado con el análisis de sentimientos. Para empezar a trabajar, tenemos un dataset con 800.000 tuits y seis variables: *sentiment* indica si el sentimiento del tuit es positivo o negativo, *id* es un identificador único del tuit, *date* indica la fecha en que fue publicado en la red social, *query* indica la consulta (si no hay mostrará "NO_QUERY"), *user* es el nombre del usuario y *text* contiene el mensaje del tuit. El dataset completo lo podéis encontrar aquí.

En esta PEC tendréis que trabajar con este dataset para procesar los textos. Los datos los tenéis en **twitter_cleaned.csv**, que está comprimido en el fichero **twitter_cleaned.zip**.

Presentación de los resultados:

Para hacer la entrega más fácil y homogénea os pedimos que organicéis el código de tal manera que **desde el fichero principal retorne todas las respuestas que se os pida en la PEC** haciendo uso de funciones que tendréis que definir en módulos. Para eso, en cada ejercicio, os indicaremos el formato que tiene que tener cada respuesta, de tal manera que ejecutando `main.py` se vaya respondiendo a toda la PEC. Por defecto, `main.py` debe ejecutar todas las funciones de la PEC mostrando cómo funcionan pero también debe permitir ejecutarlas una a una si se desea. Debéis documentarlo todo muy bien en el README para que se pueda ejecutar sin problema. Os recordamos que en el README también tenéis que indicar como ejecutar los test y comprobar la cobertura de éstos.

Control y revisión del dataset:

Cuando empezamos a trabajar en un proyecto de análisis de datos, una buena práctica es asegurarnos de que los datos son correctos. En otras palabras, es necesario hacer un análisis exploratorio inicial para detectar errores o casos especiales y tomar decisiones sobre como abordarlos. Aquí os proponemos hacer:

Ejercicio 1.1. Descomprimid el fichero twitter.zip y guardad su contenido en la carpeta data del proyecto.

Ejercicio 1.2. Leed el fichero twitter.csv y cargad el dataset como una lista de diccionarios. Cada fila del fichero original corresponderá con un diccionario siguiendo la estructura de este ejemplo:

{'sentiment': '0',

'id': '1467810369',

'date': 'Mon Apr 06 22:19:45 PDT 2009',

'query': 'NO_QUERY',

'user': '*TheSpecialOne*',

'text': "@switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer. You shoulda got David Carr of Third Day to do it. ;D"}

Mostrad por pantalla los 5 primeros registros del dataset mediante `print`.

Ejercicio 2.1. Con mucha frecuencia, los textos suelen contener elementos innecesarios o ruidosos que no aportan información relevante para el análisis. El preprocesado ayuda a eliminar esos elementos y reducir el ruido en los datos.

Realizad un preprocesado haciendo uso de **expresiones regulares** que elimine las URLs, los caracteres especiales no ASCII y los símbolos y que convierta el texto a minúsculas. Sustituid los textos originales por los modificados en el dataset del apartado anterior.

Ejercicio 2.2. Por otra parte, las stopwords son palabras comunes que no aportan un valor semántico significativo al análisis de texto. Al eliminar estas palabras, se reduce la dimensionalidad del texto y se elimina ruido adicional, permitiendo centrarse en las palabras más relevantes para el análisis. Por ejemplo, el siguiente tuit (después del preprocesado del apartado anterior): `awww that is a bummer you should got david carr of third day to do it` quedaría reducido a: `awww bummer got david carr third day` tras eliminar las stopwords.

Para este proyecto consideraremos que las stopwords son las siguientes: ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', 'her', 'hers', 'herself', 'it', 'its', 'itself',

'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', 'should', 'now']

Eliminad las stopwords de los textos de los tuits y mostrad por pantalla las 5 últimas filas.

Ejercicio 3. La técnica "Bag of Words" (BoW), se utiliza en el procesamiento del lenguaje natural (NLP) para representar y analizar textos. La idea básica detrás de la BoW es tratar un documento de texto como una "bolsa" (es decir, una colección no ordenada) de palabras, sin tener en cuenta la estructura gramatical o el orden en el que aparecen las palabras en el texto.

En la representación de la BoW, se crea un vocabulario de todas las palabras únicas que aparecen en el conjunto de documentos de texto. Luego, se cuenta el número de veces que cada palabra del vocabulario aparece en cada documento, lo que se conoce como frecuencia de término.

Obtened las frecuencias de términos de cada tuit y almacenadlas en una lista de diccionarios en la que cada diccionario indique las palabras y su frecuencia de aparición en el tuit. Obtened también un vocabulario con todas las palabras únicas del dataset y guardadlas en una lista. Mostrad por pantalla los 5 primeros elementos de la lista de diccionarios obtenida. Ordenad alfabéticamente el vocabulario y mostrad por pantalla las 10 primeras palabras.

Nota: cada elemento de la lista de frecuencias deberá tener la siguiente estructura:

{'palabra_1': número de veces que aparece la palabra_1,

'palabra_2': número de veces que aparece la palabra_2,

...

'palabra_N': número de veces que aparece la palabra_N,}

Ejercicio 4.1. Completad el dataset original añadiendo a cada registro del mismo una nueva variable con su diccionario de frecuencias de términos asociado. Mostrad el elemento 20 del dataset.

Ejercicio 4.2. Guardad el dataset procesado en formato *csv*. El nombre del fichero será *twitter_processed.csv* y se ubicará en la carpeta *data* del proyecto.

Análisis de datos:

El análisis de sentimientos juega un papel muy importante en nuestra sociedad cada vez mas digitalizada. Actualmente hay grupos de investigación que dedican todo su esfuerzo a analizar sentimientos a través de las redes sociales y estudiar tendencias de estos STOP. Debido a la gran importancia que tienen este tipo de análisis, os pedimos que utilizando el dataset anterior nos ayudéis a entender el tipo de grupos que hemos obtenido haciendo un clustering, queremos saber si los clusters obtenidos tienen sentido o no.

Para ello debéis:

Ejercicio 5. Generad una *word cloud* utilizando los tweets obtenidos en el ejercicio 4.2. Antes de generar los *word clouds* tenéis que responder las siguientes preguntas, que os ayudarán en el análisis:

1. ¿Cuántos clusters tenemos en nuestro dataset?
2. ¿Tenemos elementos vacíos en las columnas text? ¿Si es así, cuál es el porcentaje? 2.1. En caso de tener elementos nulos en la columna text, se deben eliminar antes de generar el word cloud.
3. Generar un word cloud para cada cluster.

** Definición de word cloud: En español nubes de palbras. Las nubes de palabras o nubes de etiquetas pueden utilizarse como herramientas de análisis del aprendizaje. Son representaciones visuales de un grupo de palabras utilizadas por los participantes y basadas en su frecuencia 1.

Nota 1: Los clusters los podéis encontrar en la columna sentiment.
Nota 2: Todas las respuestas anteriores se deben resolver utilizando código, mostrando el resultado en un print.

Ejercicio 6 Una vez generado el world cloud en el ejercicio anterior os pedimos que hagáis una validación de los resultados obtenidos en el apartado anterior. Para ello tenéis que generar un histograma con los valores que habéis obtenido en el ejercicio 3.

Nota: Debéis genear un histograma para cada cluster

Ejercicio 7 Analizad la *word cloud* junto con los histogramas y responded a las siguientes preguntas:

a. ¿Cuáles son las palabras más utilizadas en las críticas positivas?

b. ¿Cuáles son las palabras más utilizadas en las críticas negativas?

c. ¿Hay palabras que aparezcan tanto en las críticas positivas como en las negativas?

d. A partir de la *word cloud*, ¿qué se puede deducir sobre el sentimiento general de cada grupo?

Nota: Escribid cada respuesta en un texto corto no más de 3 líneas por respuesta.

Cobertura de los tests

La medida de la cobertura de los tests se utiliza para evaular la eficacia de los tests propuestos. En particular, sirve para determinar la calidad de los tests y determinar las partes críticas del código que no han sido testadas. Para medir este valor este valor os proponemos el uso de la herramienta `Coverage.py`. En la documentación, podéis encontrar cómo instalarla y cómo usarla.

Para evaluar los tests desarrollados en la PEC4, pedimos un mínimo del 50% de cobertura.

Criterios de corrección

Esta PEC se valorará siguiendo los siguientes criterios:

- **Funcionalidad** (5.75 puntos): Se valorará que el código implemente todo lo que se pide.
  - Ejercicio 1 (0.25 puntos)
  - Ejercicio 2 (0.75 puntos)
  - Ejercicio 3 (1 punto)
  - Ejercicio 4 (1.75 puntos)
  - Ejercicio 5 (1 puntos)
  - Ejercicio 6 (0.5 puntos)
  - Ejercicio 7 (0.5 puntos)
- **Documentación** (0.5 puntos): Todas las funciones de los ejercicios de esta PEC tendrán que estar debidamente documentadas utilizando docstrings (en el formato que prefiráis).
- **Modularidad** (1 punto): Se valorará la modularidad del código (tanto la organización del código en módulos como la creación de funciones).
- **Estilo** (0.5 puntos): El código tiene que seguir la guía de estilo de Python (PEP8), exceptuando los casos donde hacerlo complique la legibilidad del código.
- **Tests** (1.25 puntos): El código tiene que contener uno o diversas *suites* de tests que permitan comprobar que el código funciona correctamente, con un mínimo del 50% de cobertura.
- **Requeriments** (0.5 puntos): Tenéis que incluir un fichero de *requirements* que contenta la lista de librerías necesarias para ejecutar el código.
- **README** y **licencia** (0.5 puntos): Tenéis que añadir también un fichero README, que presente el proyecto y explique cómo ejecutarlo, así como la inclusión de la licencia bajo la que se distribuye el código (podéis escoger la que queráis).

Importante

Nota 1: De la misma manera que en las PECs anteriores, los criterios transversales se valorarán de manera proporcional a la parte de funcionalidad implementada.

Por ejemplo, si el código sólo implementa la mitad de la PEC y la documentación está perfecta la puntuación correspondiente a documentación será de 0.25.

Nota 2: Es imprescindible que el paquete que libréis se ejecute correctamente en la máquina virtual y que el fichero de REAMDE explique claramente cómo ejecutar el código para generar los resultados pedidos. Además en el README tiene que explicarse también cómo se ejecutarán los test y cómo se comprueba su cobertura.

Nota 3: Entregad el paquete como un único archivo .zip que contega sólo el código en el Registro de Evaluación Continua. **El código de Python tendrá que estar escrito en ficheros planos de Python.**