UNIVERSIDAD TORCUATO DI TELLA

Métodos Analíticos Aplicados a los Negocios II

Segundo Cuatrimestre 2022

TP2

Recomendaciones generales

Esta guía contiene un Trabajo Práctico de MAAN II. El mismo debe ser resuelto íntegramente en `Python`. La resolución no es simple y se espera que resolverlo tome cierto tiempo por fuera de las clases.

Recuerden que pueden consultarlos sin ningún tipo de problema y que este trabajo y los siguientes son parte de la evaluación que tendrán de la materia. Se sugiere que empiecen a resolver el TP con tiempo de modo de tener tiempo para hacer consultas y no estar apurados a último momento.

El Trabajo Práctico debe resolverse de a grupos de **2 o 3 alumnos, no más y no menos**. Las consultas podrán realizarse por mail y en los momentos asignados en clase. Tener en cuenta que si la consulta es realizada a último momento puede no ser factible responderla. **No está permitida la interacción entre grupos.**

Rompiendo códigos

En este ejercicio nos vamos a introducir en las oscuras artes de la criptografía en un contexto de escándalo internacional: descifrar (lo opuesto a cifrar) los mails de *Hillary C.*. Para ponernos en contexto, veamos qué material se entrega:

1. Un archivo de texto llamado `emails_cifrados.csv` con más de 7900 emails cifrados. Este es un archivo `.csv` que usa *Tab* (`'\t'`) como separador: la primera columna tiene el número de email, y la segunda el contenido del mismo.

2. Un script de Python llamado `auxiliares.py`.

En los archivos `.csv` (abrirlos con Notepad++ o Sublime), los textos parecen ser inentendibles. Sabemos que los mismos fueron todos cifrados usando el mismo método. Veamos detalles de cómo fueron cifrados los textos y luego cómo pueden aprovechar esto para descifrarlos:

1. Los textos fueron cifrados con un cifrado de sustitución simple monoalfabético,[1] esto quiere decir que se reemplazó en todos los textos un caracter por otro, por ejemplo la "a" podría ser el "k", la "j" podría ser la "f" y así para todos los caracteres. Como simplificación, sabemos que solo se sustituyen letras (a-z), y que los mensajes no contienen mayúsculas.

2. Sólo se cifró lo que la función `se_cifra` devuelve como `True`; de este modo, muchos de los caracteres no fueron cifrados (por ejemplo el espacio o los paréntesis o números).

3. Los emails podemos suponer que están escritos en inglés.

[1] http://es.wikipedia.org/wiki/Cifrado_por_sustitución

4. Todas las mayúsculas en los textos cifrados fueron reemplazadas por minúsculas antes de ser cifrados (se usó el método `lower`).

5. Todos los textos tienen el mismo cifrado, de modo que si se hubiera reemplazado la "a" por la "j", esto vale para todos los textos cifrados.

Por suerte para nosotros el cifrado por sustitución se puede *romper* fácilmente. Los primeros en hacerlo fueron los árabes en el siglo IX y en Europa recién en el siglo XVI se empezó a descifrar regularmente.[2] La idea detrás de la técnica usada para descifrar textos con cifrado por sustitución es muy simple. Noten que no todas las letras que aparecen en los textos de un idioma aparecen con la misma frecuencia, por ejemplo la "x" aparece poco en el español y la "a" aparece mucho. De este modo si en el texto cifrado analizamos cuál es el caracter que más aparece (supongamos la "w") y también efectuamos este análisis en textos sin cifrar en español y vemos cuál es el caracter que más aparece (supongamos la "a"), es de esperar que estos dos caracteres se correspondan (que la "w" del texto cifrado sea en realidad una "a"). Esto se puede pensar también para la segunda letra que más aparece, para la tercera y así.[3]

Entonces, en pocas palabras, lo que deberán hacer es establecer una correspondencia entre caracteres, tratando de encontrar cuál es el caracter utilizado en cada caso. Para ello, los pasos a seguir son los siguientes:

- Calcular la frecuencia de caracteres en los mensajes cifrados. Para ello, recorrer los mensajes, contar la aparición de cada caracter y luego ordenarlo por cantidad de apariciones de mayor a menor.

- Utilizar alguna fuente que permita identificar la frecuencia de apariciones de caracteres en Inglés en general. Para ello, consideramos una fuente estadística de apariciones de caracteres, como por ejemplo frecuencias en inglés.

- Una vez que tenemos los caracteres de ambos ordenados, podemos establecer la correspondencia entre los mismos intentando obtener relación inversa utilizada para codificar los mensajes.

- Luego, tomar uno de los mensajes cifrados y reemplazar en función de la correspondencia que identificamos.

- La técnica de análisis de frecuencias no anda 100 % perfecto, pero si anda lo suficientemente bien para romper los cifrados. Por ejemplo puede ser que reemplazando vean que una palabra les quede mal escrita (por ejemplo *lapiceru* en lugar de *lapicera*). Si ven esto, es claro que la "u" es en realidad una "a"... ¡Estas correcciones háganlas "a mano"! (¡o con código Python mejor!). Debido a que las fuentes utilizadas para establecer la correspondencia entre caracteres viene de distintos contextos, es posible que se requieran ciertas modificaciones manuales para ajustar las diferencias. Es de esperar que el porcentaje de aciertos en la correspondencia sea alrededor del 50 %. *Sugerencia: notar que los mensajes son emails, y empezar por palabras cortas identificables para las correcciones.*

[2] A algunos les costó creerlo posible. Por ejemplo, el Rey Felipe II de España, fue tan lejos como para pedir al Vaticano que un famoso descifrador francés fuese juzgado porque Felipe afirmaba que la única forma de descifrar los textos españoles era con ayuda del diablo.

[3] http://es.wikipedia.org/wiki/Análisis_de_frecuencias

- Por último, una vez que se ha logrado entender el primer mensaje, se pueden decodificar los restantes 7900.

Tip para los usuarios de Windows: muchas veces Python y caracteres raros no se llevan bien con la consola de Windows (que usa un encoding muy malo... MUY MUY MUUUY MALO), por eso les va a ser mucho más fácil guardar las salidas en archivos de texto y verlas con Notepad++ (o Sublime) antes que imprimirlas en pantalla.

Evaluación

La resolución del trabajo consiste en completar el código agregando aquellas funciones auxiliares que consideren necesarias. A continuación se provee un detalle por etapas, con la explicación de las funciones provistas por la cátedra. En todos los casos se presenta un ejemplo de input / output.Se pide:

1. **(1.5 puntos)** Lectura de archivo de emails cifrados mediante la función `leer_mails`. Dado el nombre de un archivo en formato `csv`, con dos columnas y separado por *tab* (`\t`), se debe retornar una lista que posea tantas posiciones como líneas en el archivo, donde el mensaje de la i-ésima línea se corresponde con la i-ésima posición de la lista.

   Input: `emails_test.csv` (mostrado en múltiples líneas)

   ```
   1 wlsxzkkghgat w.k. taezncdalc fh kczca szka lf. h-2015-04841 tfs lf.
         s05739545 tzca: 05/13/2015 kczca taec. - enftwsat cf vfwka kaxasc
         jalyvzug sfdd.
   2 abcdefghijklmnopqrstuvwxyz
   ```

   Output: variable `emails_cifrados` en la función `main` (mostrado en múltiples líneas)

   ```
   [
     'wlsxzkkghgat w.k. taezncdalc fh kczca szka lf. h-2015-04841 tfs lf.
        s05739545 tzca: 05/13/2015 kczca taec. - enftwsat cf vfwka kaxasc
        jalyvzug sfdd.\n',
     'abcdefghijklmnopqrstuvwxyz'
   ]
   ```

2. **(3 puntos)** Obtención y ordenamiento de frecuencias (tanto de los mails cifrados como la frecuencia estática). Estas tareas se realizan mediante dos funciones.

   Por un lado, primero se calcula la frecuencia de los caracteres que aparecen en los mensajes cifrados (recordar, aquellos con `se_cifra(c)` es `True`) mediante la función `obtener_frecuencia_caracteres(dataset)`.

   Input: `emails_cifrados` del punto anterior.

   Output: un diccionario con claves de tipo `string` (los caracteres) y significado `int` (cantidad de apariciones en todos los textos). Corresponde a la variable `freq_cifrado` en la función `main` (mostrado en múltiples líneas)

   ```
   {'w': 5, 'l': 6, 's': 8, 'x': 3, 'z': 8, 'k': 9, 'g': 4, 'h': 4, 'a': 14, 't': 8, 'e': 4,
    'n': 3, 'c': 11, 'd': 4, 'f': 9, 'v': 3, 'j': 2, 'y': 2, 'u': 2, 'b': 1, 'i': 1, 'm': 1,
    'o': 1, 'p': 1, 'q': 1, 'r': 1}
   ```

Luego, la función `ordenar_por_frecuencia(freq_dict)` tomo como input este diccionario y retorna una lista de tuplas, donde cada tupla corresponde a un item `(c,n)` del diccionario, y los mismos se encuentran ordenados de forma decreciente en función de `n`.

Input: `freq_cifrado` del punto anterior.

Output: una lista de tuplas, cada tupla un ítem de `freq_dict`, ordenada de forma decreciente en función de la segunda componente de la tupla. Corresponde a la variable `orden_cifrado` en la función `main`.

```
[('a', 14), ('c', 11), ('k', 9), ('f', 9), ('s', 8), ('z', 8), ('t', 8), ('l', 6), ('w', 5),
 ('g', 4), ('h', 4), ('e', 4), ('d', 4), ('x', 3), ('n', 3), ('v', 3), ('j', 2), ('y', 2),
 ('u', 2), ('b', 1), ('i', 1), ('m', 1), ('o', 1), ('p', 1), ('q', 1), ('r', 1)]
```

3. **(2 puntos)** Inferencia automática de correspondencia y posterior corrección manual. Esta tarea también se realiza mediante tres funciones. Por un lado, ya se provee la función `obtener_frecuencia_estatica()` que retorna por defecto una lista de caracteres, ya ordenados por frecuencia de aparición, en base a análisis realizados de textos en Inglés. Esto se corresponde con la variable `orden_sin_cifrar` en la función `main`.

   El siguiente paso es establecer la correspondencia entre los caracteres ordenados por frecuencia en el texto cifrado y los caracteres ordenados por frecuencia del idioma Inglés. Esta tarea se realiza en la función:

   ```
   def inferir_correspondencia(orden_cifrado, orden_sin_cifrar)
   ```

   que tiene el siguiente comportamiento:

   Input: `orden_cifrado` corresponde a `freq_cifrado` del punto anterior, y `orden_sin_cifrar` a los caracteres ordenados de forma estática. Notar que uno de ellos es una lista de tuplas, mientras que el otro una lista de caracteres.

   Output: retorna un diccionario `d`, con claves y significados ambos `strings`. Las claves representan un caracter en el texto cifrado, y dada una clave `k` del diccionario `d[k]` indica el caracter por el que debe ser reemplazado para descifrar el texto. Continuando con el ejemplo, el output de la función debería ser el siguiente:

   ```
   {'a': 'e', 'c': 't', 'k': 'a', 'f': 'o', 's': 'i', 'z': 'n', 't': 's', 'l': 'h', 'w': 'r',
    'g': 'd', 'h': 'l', 'e': 'c', 'd': 'u', 'x': 'm', 'n': 'w', 'v': 'f', 'j': 'g', 'y': 'y',
    'u': 'p', 'b': 'b', 'i': 'v', 'm': 'k', 'o': 'j', 'p': 'x', 'q': 'q', 'r': 'z'}
   ```

   Notar que la correspondencia depende de las frecuencias inferidas, que a su vez dependen del input de mensajes cifrados. Esto se corresponde con la variable `correspondencia_inf` en el mail.

4. **(1 punto)** Traducción del mensaje usando la correspondencia entre caracteres cifrados y no cifrados. Esta etapa se implementa mediante la función

   ```
   def descifrar_mensaje(mensaje, correspondencia):
   ```

   Esta funcíon toma un mensaje cifrado, la correspondencia inferida en el paso anterior y aplica el reemplazo correspondiente. Notar por la naturaleza del problema, tal como se mencionó anteriormente en el enunciado, esta traducción seguramente todavía no sea completa y posea errores.

La corrección de la correspondencia será un proceso manual, que consiste en modificar la correspondencia inferida automáticamente mediante un análisis secuencial de los mensajes traducidos. Esto, en resumen, es aplicar redefiniciones al diccionario `correspondencia_inf` obtenido en el punto anterior. Los mismos serán encapsulados en la función

```
def corregir_correspondencia(correspondencia):
```

5. **(1.5 puntos)** Escritura de mensajes traducidos en un nuevo archivo. Una vez corregida la correspondencia, el siguiente paso es aplicar de forma automática la traducción de todos los mensajes y escribir los mismos a un archivo de output con el mismo formato que el de entrada.
6. **(1 puntos)** Comentarios en el código y armado de funciones auxiliares.

Modalidad de entrega

Además del código con las implementaciones es posible entregar un mini informe de no más de tres carillas en donde detallen, en caso de considerarlo necesario, decisiones que tomaron para resolver uno o más ejercicios y que ayuden a entender la estrategia de resolución.

Fechas de entrega

Formato Electrónico: **viernes 14 de octubre, hasta las 23:59 hs.**, enviando el trabajo (informe + código) a la dirección `maan2utdt@gmail.com`. El subject del email debe comenzar con el texto [TP2 MAAN II] seguido de la lista de apellidos de los integrantes del grupo. Todos los integrantes del grupo deben estar copiados en el email.

Importante: El horario es estricto. Los correos recibidos después de la hora indicada serán considerados re-entrega.