

Tratamiento Inteligente de datos (TID)

Prácticas de la asignatura
2018-2019

En colaboración con:



ugr

Universidad
de Granada

Participantes

Alejandro Campoy Nieves: alejandroac79@correo.ugr.es

Gema Correa Fernández: gecorrea@correo.ugr.es

Luis Gallego Quero: lgag94@correo.ugr.es

Jonathan Martínez Valera: jmv742@correo.ugr.es

Andrea Morales Garzón: andreamgmg@correo.ugr.es

Índice

| | |
|---|----------|
| 1. Comprender el problema a resolver | 1 |
| 2. Preprocesamiento de datos | 1 |
| 2.1. Lectura de datos | 1 |
| 2.2. Falta de datos, categorización, normalización, reducción de dimensionalidad. | 5 |
| Procesar datos | 5 |
| Eliminar columna del ID. | 5 |
| Creación del corpus | 5 |
| Eliminar signos de puntuación. | 6 |
| Stopwords. | 6 |
| Stemming | 7 |

Índice de figuras

Índice de cuadros

1. Comprender el problema a resolver

Para la realización y aplicación de las técnicas explicadas a lo largo del curso, hemos seleccionado un *dataset* proporcionado por *UCI Machine Learning Repository*. En concreto, hemos escogido **Drug Review Dataset**, una exhaustiva base de datos de medicamentos organizada por relevancia para medicamentos específicos. El conjunto de datos proporciona revisiones de pacientes sobre medicamentos específicos junto con las condiciones relacionadas. Además, las revisiones se agrupan en informes sobre tres aspectos: beneficios, efectos secundarios y comentarios generales. De igual modo, las calificaciones están disponibles con respecto a la satisfacción general, así como una calificación de efectos secundarios y de eficacia de 5 pasos. Los datos se obtuvieron rastreando los sitios de revisión farmacéutica en línea.

El objetivo principal del estudio es:

- Realizar un análisis de sentimientos en relación con la experiencia en el uso de dichos medicamentos, como por ejemplo la efectividad, efectos secundarios. . .
- Compatibilizar dicho modelo de datos con otros conjuntos de datos aportados en: **Drugs.com**

En este proyecto nos centraremos en el **análisis y experiencia de los usuarios** en el uso de los distintos medicamentos.

Las características de este conjunto de datos vienen descritas en la siguiente tabla:

| | | | | | |
|-------------------|--|-----------------|------|---------------------|----------|
| DataSet | | Number of | | | |
| Characteristics: | Multivariate, Text | Instances: | 4143 | Area: | N/A |
| Attribute | Integer | Number of | 8 | Date Donated | 2018-10- |
| Characteristics: | | Attributes: | | | 02 |
| Associated Tasks: | Classification, Regression, Clustering | Missing Values? | N/A | Number of Web Hits: | 7001 |

Los datos se dividen en un conjunto train (75 %) y otro conjunto test (25 %) y se almacenan en dos archivos.tsv (tab-separated-values), respectivamente. Los atributos que tenemos en este dataset son:

1. **urlDrugName** (categorical): nombre de la droga
2. **condition** (categorical): nombre de la condición
3. **benefitsReview** (text): paciente sobre beneficios
4. **sideEffectsReview** (text): paciente sobre los efectos secundarios
5. **commentsReview** (text): comentario general del paciente
6. **rating** (numerical): clasificación de paciente de 10 estrellas
7. **sideEffects** (categorical): clasificación de 5 pasos de efectos secundarios
8. **effectiveness** (categorical): clasificación de efectividad de 5 pasos

2. Preprocesamiento de datos

Para poder analizar el dataset y realizar el preprocesamiento al mismo, lo primero que se va hacer es leer tanto el conjunto de datos train como de test. Primero, leeremos los datos con los que se va a entrenar y luego los datos test.

2.1. Lectura de datos

A continuación, leemos nuestro dataset train y test:

```
# Lectura de datos train
datos_train <- read.table("datos/drugLibTrain_raw.tsv", sep="\t", comment.char="",
                          quote = "\"", header=TRUE)
head(datos_train, 5) # visualizar las 5 primeras filas
```

```
##      X      urlDrugName rating      effectiveness      sideEffects
## 1 2202      enalapril      4      Highly Effective Mild Side Effects
## 2 3117 ortho-tri-cyclen      1      Highly Effective Severe Side Effects
## 3 1146      ponstel      10      Highly Effective      No Side Effects
## 4 3947      prilosec      3 Marginally Effective Mild Side Effects
## 5 1951      lyrica      2 Marginally Effective Severe Side Effects
##
##      condition
## 1 management of congestive heart failure
## 2      birth prevention
## 3      menstrual cramps
## 4      acid reflux
## 5      fibromyalgia
##
## 1
## 2
## 3
## 4 The acid reflux went away for a few months after just a few days of being on the drug. The heartbu
## 5
##
## 1      cough, hypotension , proteinuria, impo
## 2 Heavy Cycle, Cramps, Hot Flashes, Fatigue, Long Lasting Cycles. It's only been 5 1/2 months, but i
## 3
## 4
## 5
##
## 1
## 2
## 3 I took 2 pills at the onset of my menstrual cramps and then every 8-12 hours took 1 pill as needed
## 4
## 5
```

```
summary(datos_train) # información sobre los datos
```

```
##      X      urlDrugName      rating
## Min.   : 0      lexapro : 63      Min.   : 1.000
## 1st Qu.:1062      prozac  : 46      1st Qu.: 5.000
## Median :2092      retin-a : 45      Median : 8.000
## Mean   :2081      zoloft  : 45      Mean   : 7.006
## 3rd Qu.:3092      paxil   : 38      3rd Qu.: 9.000
## Max.   :4161      propecia: 38      Max.   :10.000
##      (Other) :2832
##      effectiveness      sideEffects
## Considerably Effective: 928      Extremely Severe Side Effects: 175
## Highly Effective      :1330      Mild Side Effects      :1019
## Ineffective           : 247      Moderate Side Effects   : 614
## Marginally Effective  : 187      No Side Effects         : 930
## Moderately Effective  : 415      Severe Side Effects     : 369
##
##
```

```

##              condition
## depression      : 236
## acne            : 165
## anxiety         : 63
## insomnia        : 54
## birth control   : 49
## high blood pressure: 42
## (Other)         :2498
##
## none
## None
## NONE
## None.
## The treatment benefits were marginal at best. Mood neither improved nor deteriorated, and anxiety w
## Before the use of vagifem tablets, I had to endure a series of urinary infections after sometimes p
## (Other)
##              sideEffectsReview      commentsReview
## none              : 112      n/a              : 7
## None              : 73      none              : 6
## None.             : 19      None              : 4
## No side effects.   : 9       .              : 3
## There were no side effects.: 6      One tablet once a day: 3
## no side effects    : 5       (Other)         :3083
## (Other)            :2883      NA's              : 1

```

```

View(datos_train)      # vista de la tabla

# Lectura de datos test
datos_test <- read.table("./datos/drugLibTest_raw.tsv", sep="\t", comment.char="",
                        quote = "\"", header=TRUE)
head(datos_test, 5) # visualizar las 5 primeras filas

```

```

##      X urlDrugName rating      effectiveness      sideEffects
## 1 1366      biacin      9 Considerably Effective Mild Side Effects
## 2 3724    lamictal      9      Highly Effective Mild Side Effects
## 3 3824    depakene      4 Moderately Effective Severe Side Effects
## 4 969      sarafem     10      Highly Effective No Side Effects
## 5 696      accutane     10      Highly Effective Mild Side Effects
##              condition
## 1      sinus infection
## 2      bipolar disorder
## 3      bipolar disorder
## 4 bi-polar / anxiety
## 5      nodular acne
##
## 1
## 2 Lamictal stabilized my serious mood swings. One minute I was clawing up the walls in pure mania, t
## 3
## 4
## 5
##
## 1
## 2 Drowsiness, a bit of mental numbness. If you take too much, you will feel sedated. Since you have
## 3
## 4

```

```
## 5
##
## 1
## 2
## 3 Depakote was prescribed to me by a Kaiser psychiatrist in Pleasant Hill, CA in 2006. The medication
## 4
## 5
```

```
summary(datos_test) # información sobre los datos
```

```
##           X           urlDrugName           rating
## Min.      : 1.0      paxil       : 20      Min.      : 1.000
## 1st Qu.: 968.2      effexor-xr: 17      1st Qu.: 5.000
## Median :2048.0      accutane   : 16      Median : 8.000
## Mean    :2085.4      synthroid : 15      Mean    : 6.767
## 3rd Qu.:3199.8      differin  : 13      3rd Qu.: 9.000
## Max.    :4157.0      effexor   : 13      Max.    :10.000
##              (Other)   :942
##              effectiveness      sideEffects
## Considerably Effective:310      Extremely Severe Side Effects: 80
## Highly Effective      :411      Mild Side Effects              :330
## Ineffective           : 82      Moderate Side Effects          :236
## Marginally Effective  : 76      No Side Effects                :268
## Moderately Effective  :157      Severe Side Effects            :122
##
##
##              condition
## depression        : 66
## acne              : 46
## anxiety            : 27
## insomnia           : 21
## high blood pressure: 20
## birth control      : 19
## (Other)            :837
##
## none
## None
## elevation of mood and clarity of thought. Progress stalled out at 300 mg, but with increase to 450
## I've only been on it for a week but I've noticed a change already. I am more awake and it seems as if
## The benefits of using Tretinoin were great. First of all I noticed that my skin started glowing and
## !0 years after spinal stenosis, scar-tissue and additional narrowing of nerve canals causes severe in
## (Other)
##
## none
## None
## None.
## none at 300 mg. Possible tinnitus from increase to 450 mg, not evaluated by an audiologist yet though
## dryness
## luckily I did not notice any negative side effects. The positive effects that I noticed out weighed
## (Other)
##
## Initial treatment included therapy and Lexapro in addition to Wellbutrin XL. Now only on the Wellbutrin
## My doctor added Abilify to my 60 mg of Cymbalta because I was feeling really fatigued and unable to
## My treatment details are as follows: I Used Avita (Tretinoin) every night after cleansing my face.
## none
```



```
## see above
## 'Heart failure' probably due to muscle wastage, as concurrentl seen in external muscles, as a resul
## (Other)
View(datos_test)    # vista de la tabla
```

2.2. Falta de datos, categorización, normalización, reducción de dimensionalidad.

Procesar datos

Eliminar columna del ID.

Como ya se ha comentado previamente, al conjunto de datos utilizado se le ha añadido de forma automática una novena columna, que representa un ID para cada uno de los datos con los que estamos trabajando. Como no nos aporta información, hemos decidido quitarla directamente del dataframe. Esta columna se corresponde con la primera columna del dataframe, por lo cuál, debemos eliminar la columna que se corresponde con la posición 1.

```
datos_test = datos_test[-1]
```

Una vez eliminada la columna anterior, ya podemos continuar con el procesamiento de los datos.

Primero tenemos que usar la librería que procese los datos de tipo texto en R. La más conocida se llama tm. Si no la tenemos instalada en R, tenemos que hacerlo previamente. Para eso hacemos lo siguiente:

```
# install.packages("tm")
# install.packages("SnowballC")

library("tm")
```

```
## Warning: package 'tm' was built under R version 3.4.4
## Loading required package: NLP
## Warning: package 'NLP' was built under R version 3.4.4
library("SnowballC")
```

```
## Warning: package 'SnowballC' was built under R version 3.4.4
```

Creación del corpus

Para poder obtener la estructura con la que vamos a procesar nuestra información, debemos obtener un vector con documentos. En nuestro caso, cada uno de los documentos se corresponde con una opinión sobre un fármaco. Para ello, primero debemos obtener un vector con todas las opiniones (o lo que es lo mismo, con todas). Para ello debemos extraerlo del dataframe en forma de vector.

Sólo nos faltaría convertir cada elemento del vector al formato de documento. Podemos usar la función VectorSource para hacer esta conversión.

```
# Nos quedamos con la única columna del dataset que nos interesa. Necesitamos obtenerla en forma de vec
benefits_review_data = as.vector(datos_test$benefitsReview)

# Lo convertimos en la estructura de documento, y lo guardamos ya en el corpus que lo vamos a utilizar.
corpus = (VectorSource(benefits_review_data))

# Creamos el propio corpus
```

```
corpus <- Corpus(corpus)
#summary(corpus)
```

Podemos ver que funciona accediendo a uno cualquiera, por ejemplo, vamos a acceder al cuarto comentario almacenado.

```
# P Si nos fijamos en el contenido, vemos que tiene signos de puntuación y exclamación.
inspect(corpus[4])
```

```
## <<SimpleCorpus>>
## Metadata: corpus specific: 1, document level (indexed): 0
## Content: documents: 1
##
## [1] It controlls my mood swings. It helps me think before i act or speak. It controlls amxiety. IT FREE
corpus[[4]]$content
```

```
## [1] "It controlls my mood swings. It helps me think before i act or speak. It controlls amxiety. IT FREE"
```

Eliminar signos de puntuación.

Como hemos podido ver en el documento que se ha mostrado por pantalla, en él se aprecia el uso de signos de puntuación y exclamación. En un principio, no tiene sentido en *Data Mining* contemplar los signos de puntuación, ya que no nos van a aportar información. Por ello, los quitamos, como se puede ver a continuación.

```
# 4. Una vez que tenemos el corpus creado, continuamos con el procesamiento.
corpus <- tm_map(corpus, content_transformer(removePunctuation))
```

```
## Warning in tm_map.SimpleCorpus(corpus,
## content_transformer(removePunctuation)): transformation drops documents
```

Si volvemos a mostrar la opinión número cuatro, vemos como todos los signos han desaparecido. De hecho, podemos inspeccionar el corpus, y se ve como todos los signos de puntuación, exclamación y derivados ya no están.

```
inspect(corpus[4])
```

```
## <<SimpleCorpus>>
## Metadata: corpus specific: 1, document level (indexed): 0
## Content: documents: 1
##
## [1] It controlls my mood swings It helps me think before i act or speak It controlls amxiety IT FREE
```

Stopwords.

En cualquier idioma, hay palabras tan tan tan comunes que no nos aportan información relevante. Por ejemplo, en español, las palabras “la”, “a”, “en”, “de” son ejemplos de lo que se conoce como Stopwords. Este tipo de palabras, tenemos que suprimirlas de nuestro corpus. Como el contenido del corpus está en inglés, debemos especificar el idioma correcto para que nos elimine del corpus las palabras adecuadas.

```
corpus <- tm_map(corpus, content_transformer(removeWords),
  stopwords("english"))
```

```
## Warning in tm_map.SimpleCorpus(corpus, content_transformer(removeWords), :
## transformation drops documents
```

Sin embargo, tenemos otro problema, y es que no todas las stopwords se han borrado. Sí se han borrado todas las stopwords cuyas letras que la componen están en minúscula, pero no si una de las letras están en mayúscula. Por ejemplo: todas las “the” se han borrado, pero no las “The”. Esto nos obliga a tener que pasar todos los caracteres del texto a letras minúscula, y a repetir de nuevo el proceso que elimina las stopwords.

```
corpus <- tm_map(corpus, content_transformer(tolower))
```

```
## Warning in tm_map.SimpleCorpus(corpus, content_transformer(tolower)):  
## transformation drops documents
```

```
inspect(corpus[1])
```

```
## <<SimpleCorpus>>  
## Metadata: corpus specific: 1, document level (indexed): 0  
## Content: documents: 1  
##  
## [1] the antibiotic may destroyed bacteria causing sinus infection but may also caused virus
```

```
corpus <- tm_map(corpus, content_transformer(removeWords), stopwords("english"))
```

```
## Warning in tm_map.SimpleCorpus(corpus, content_transformer(removeWords), :  
## transformation drops documents
```

Ahora ya hemos eliminado las stopwords de forma correcta.

Stemming

El siguiente paso consiste en reducir el número de palabras totales con las que estamos trabajando. En este caso, se trata de reducir aquellas que no nos aportan nada relevante a lo que ya tenemos. En la columna con la que estamos trabajando en este dataframe, se repite una gran cantidad de veces la palabra “benefit”, al igual que “benefits”.

Sin embargo, realizar el análisis de nuestros datos con ambas palabras no tiene gran relevancia, ya que una no aporta nada respecto a la otra. Este es un ejemplo del tipo de casos que se nos dan en nuestro dataset; por ejemplo. Igual ocurre con “reduce” y “reduced”, por ejemplo. Este tipo de situaciones son las que intentamos corregir con este paso.

```
inspect(corpus[340])
```

```
## <<SimpleCorpus>>  
## Metadata: corpus specific: 1, document level (indexed): 0  
## Content: documents: 1  
##  
## [1] benefit drug yes work crazy locked mental ward usage became normal crazy againhow
```

```
inspect(corpus[272])
```

```
## <<SimpleCorpus>>  
## Metadata: corpus specific: 1, document level (indexed): 0  
## Content: documents: 1  
##  
## [1] benefits protonix greatly reduced incidence heartburn spams esophagus
```

```
corpus <- tm_map(corpus, stemDocument)
```

```
## Warning in tm_map.SimpleCorpus(corpus, stemDocument): transformation drops  
## documents
```

```
inspect(corpus[340])
```

```
## <<SimpleCorpus>>
## Metadata: corpus specific: 1, document level (indexed): 0
## Content: documents: 1
##
## [1] benefit drug yes work crazi lock mental ward usag becam normal crazi againhowev treatment plan g
```

```
inspect(corpus[272])
```

```
## <<SimpleCorpus>>
## Metadata: corpus specific: 1, document level (indexed): 0
## Content: documents: 1
##
## [1] benefit protonix great reduc incid heartburn spam esophagus
```

Si ahora volvemos a mostrar el contenido de dichas opiniones, podemos ver como el stemming se ha hecho efectivo: donde ponía *benefits* ahora pone *benefit*, y la misma operación se ha llevado a cabo para todos los comentarios en el dataset.

```
inspect(corpus[340])
```

```
## <<SimpleCorpus>>
## Metadata: corpus specific: 1, document level (indexed): 0
## Content: documents: 1
##
## [1] benefit drug yes work crazi lock mental ward usag becam normal crazi againhowev treatment plan g
```

```
inspect(corpus[272])
```

```
## <<SimpleCorpus>>
## Metadata: corpus specific: 1, document level (indexed): 0
## Content: documents: 1
##
## [1] benefit protonix great reduc incid heartburn spam esophagus
```