ITESO

Text Mining

PAP Modelación Matemática para el desarrollo de planes y proyectos de negocio

Sara Eugenia Rodríguez Reyes

17/11/2015

INTRODUCCIÓN

Text Mining es el proceso de derivar la información de alta calidad a partir de texto. Esta se obtiene a través de la elaboración de patrones y tendencias a través de medios tales como el aprendizaje estadístico.

Implica el proceso de estructuración del texto de entrada (análisis, adición y eliminación de rasgos lingüísticos), derivando patrones dentro de los datos estructurados y, finalmente, la evaluación e interpretación.

Para este trabajo se analizarán dos noticas sobre la muerte de Bob Marley, ambos documentos en el idioma inglés.

Documento 1:

http://www.nydailynews.com/news/world/bob-marley-died-cancer-1981-article-1.2100349

Documento 2:

http://www.theguardian.com/theguardian/1981/may/12/1

```
#Cargar librerías
library (tm)
library (SnowballC)
library (RColorBrewer)
library (ggplot2)
library (wordcloud)
library (biclust)
library (cluster)
library (igraph)
library (fpc)
install.packages("Rcampdf", repos = "http://datacube.wu.ac.at/", type =
"source")
#Cargar Datos
#Los archivos están guardados en una carpeta llamada "texts" en C:/
cname <- file.path("C:", "texts")
cname
dir(cname)
docs <- Corpus(DirSource(cname))
summary(docs)
```

Preprocesamiento

Una vez que los documentos están cargados corretamente, se van a pre-procesar los textos. Este paso permite eliminar números, mayúsculas, palabras comunes, puntuación, etc. Esto puede ser un poco lento y exigente, pero vale la pena al final, en términos de análisis de mejor calidad.

```
docs <- tm_map(docs, removePunctuation) #quitar puntuación
for(j in seq(docs))
{
  docs[[j]] <- gsub("-", " ", docs[[j]])
}
docs <- tm_map(docs, removeNumbers) #quitar números
docs <- tm_map(docs, tolower) #pasar mayúsculas a minúsculas
```

Extracción de " palabras comunes " que por lo general no tienen valor analítico.
En cada texto, hay un montón de éstas, y las palabras sin interés (a, y, también, la, etc.). Tales palabras son frecuentes, por su naturaleza, y confunden el análisis si permanecen en el texto.

```
length(stopwords("english"))
stopwords("english")
docs <- tm_map(docs, removeWords, stopwords("english"))
```

Remover palabras comunes con terminaciones (“ing”, “es”, “s”). A esto se le llama “stemming”; sirve para que la palabra sea reconocible para la computadora, a pesar de si tiene muchas variedades de terminaciones en el texto original.

```
docs <- tm map(docs, stemDocument)
```

Quitar espacios en blanco innecesarios. Lo hecho anteriormente, deja a los documentos con muchos espacios en blanco. Esto es resultado de todos los espacios que no fueron removidos de las palabras que fueron eliminadas.

```
docs <- tm_map(docs, stripWhitespace)
docs <- tm_map(docs, PlainTextDocument)
```

Exploración de datos

Se crea un documento en forma de matriz, luego se traspone

```
dtm <- DocumentTermMatrix(docs)
tdm <- TermDocumentMatrix(docs)
```

Se organizan los términos por su frecuencia.

```
freq <- colSums(as.matrix(dtm))
length(freq)
ord <- order(freq)
```

Remover términos dispersos. Se hace una matriz con el 10% de espacio vacío.

```
dtms <- removeSparseTerms(dtm, 0.1)
inspect(dtms)
```

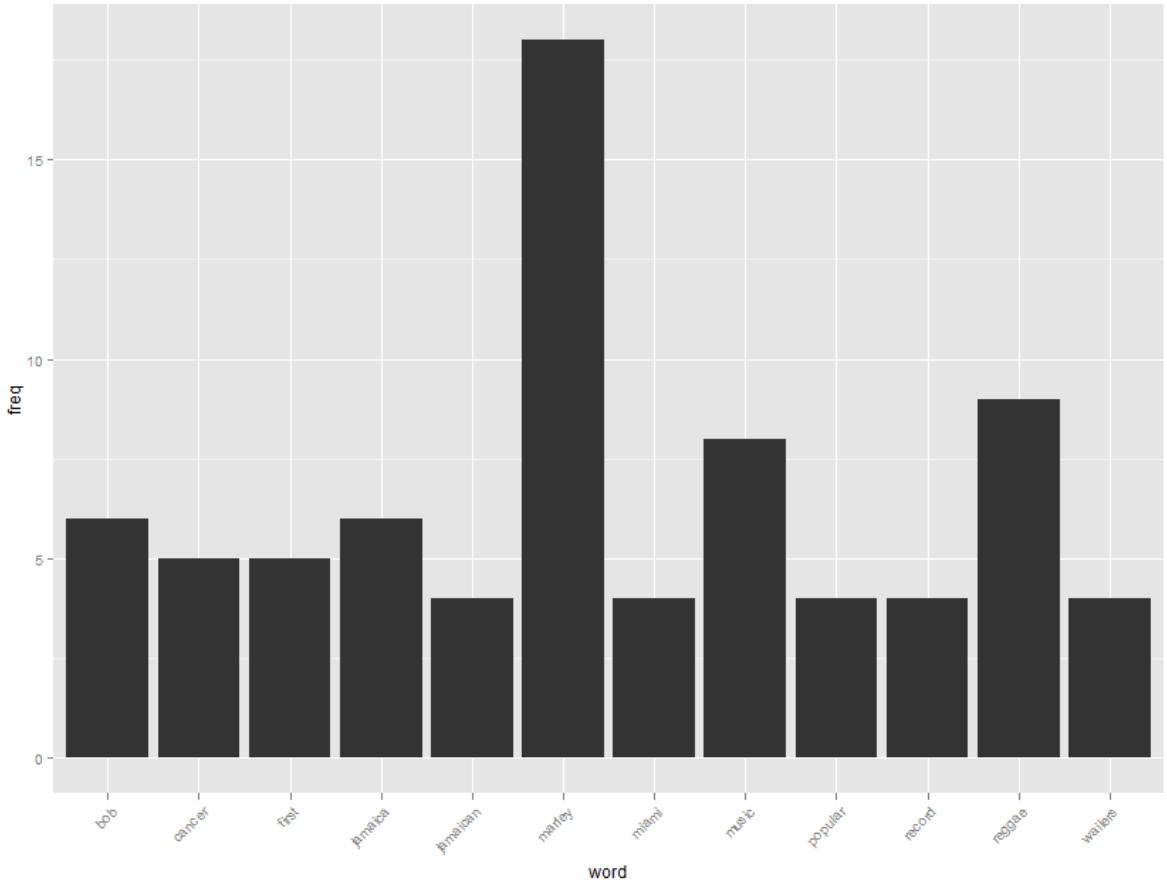
Frecuencia de las palabras

```
freq[head(ord)]
freq[tail(ord)]
head(table(freq), 20)
tail(table(freq), 20)
freq <- colSums(as.matrix(dtms))#Tabla de los términos elegidos cuando se
removieron los términos dispersos
wf <- data.frame(word=names(freq), freq=freq)
head(wf)
```

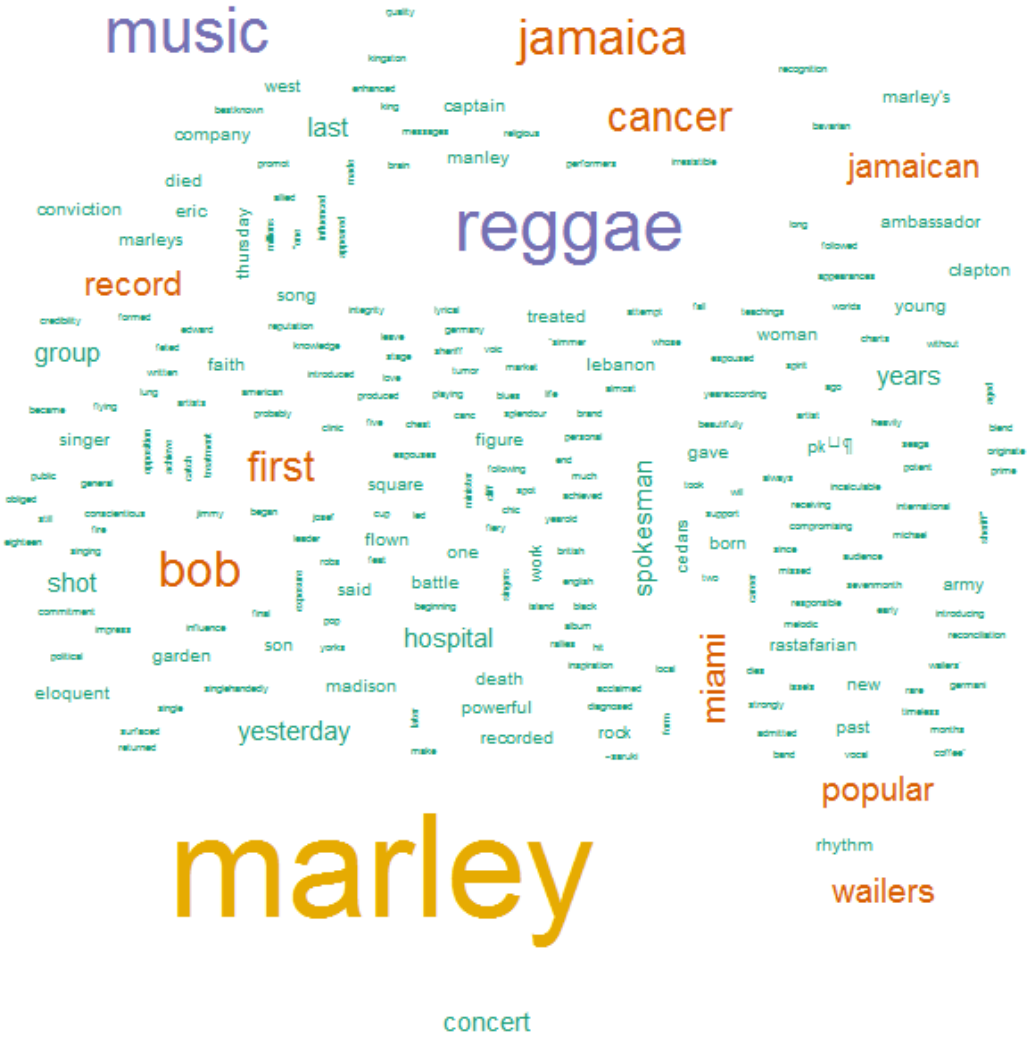
```
            word freq
marley    marley   18
reggae    reggae    9
music      music    8
bob          bob    6
jamaica  jamaica    6
cancer    cancer    5
```

La palabra más utilizada es Marley, seguida de reggae, music, Bob, Jamaica, Cáncer

```
p <- ggplot(subset(wf, freq>3), aes(word, freq))
p <- p + geom_bar(stat="identity")
p <- p + theme(axis.text.x=element_text(angle=45, hjust=1))
p
```



```
set.seed(142)
wordcloud(names(freq), freq, min.freq=20, scale=c(5, .1), colors=brewer.pal(6,
"Dark2"))
```



Clustering por similitud de palabras

Primero se calcula la distancia entre palabras para luego agruparlas de acuerdo a su similitud.

```
d <- dist(t(dtmss), method="euclidian")
fit <- hclust(d=d, method="ward")
fit
plot.new()
plot(fit, hang=-1)
groups <- cutree(fit, k=5)
rect.hclust(fit, k=5, border="red")
```