03/22/2020

Carlos Alberto Cordero Robles

**aplicar k-means a un dataset con 3 tipos de textos**

**Introducción**

En esta tarea se analizará un texto con frases relacionadas a tres temas los cuales serán: cerveza, filosofía e ingenieros. Se planea usar clustering con Kmeans para identificar de que tipo son.

El dataset será un archivo TXT el cual fue generado compilando frases de [1,2,3] para luego crear un corpus. Se analizará el listado de palabras para identificar las que no den valor agregado al análisis y así agregarlas a la lista de stopwords. Posteriormente se aplicará una técnica de inverted matrix para darle peso a cada palabra en cada frase y al final se aplicará Kmeans para hacer clusters y clasificar.

**Dataset**

El dataset se creó copiando 10 frases de cada tipo (cerveza, filosofía e ingenieros) de las páginas [1,2,3]. Las frases no son mayores de 20 palabras. Esto se hizo para que la longitud de las frases no sean un factor que afecte el resultado. Se eligió esos tres temas ya que de alguna manera son de mi interés y también porque no son temas tan comunes entre sí pero si hay una pequeña relación entre ellos, por lo tanto se espera que solo haya algunos errores en la clasificación.

**Proceso**

1.- integrar una dataset con 3 tipos de textos, breves.

Se copiaron y pegaron en un TXT diez frases de cada tipo. Haciendo uso de un editor de texto elimine los signos de puntuación para quedarnos con las puras palabras, el resultado fue el siguiente.

|  |
| --- |
| La cerveza es la prueba de que Dios nos ama y quiere que seamos felices  La cerveza es intelectual Qué lástima que tantos idiotas la beban  A mucha gente no le gusta el sabor de la cerveza sin embargo eso no es más que un prejuicio  Denme una mujer que ame la cerveza y conquistaré el mundo  Un hombre que miente sobre la cerveza hace enemigos  La mejor cerveza está donde van a beber los monje  Nada sabe mejor que una cerveza fría en una bella tarde con nada más que esperar sino más de lo mismo  Qué importa si el tiempo avanza si hoy estoy tomándome una cerveza  La cerveza es la causa y solución de todos los problemas de la vida  Dios tiene una voz parda suave y plena como una cerveza  La cosa más difícil es conocernos a nosotros mismos la más fácil es hablar mal de los demás  No puedo enseñar nada a nadie Solo puedo hacerles pensar  No juzgamos a las personas que amamos  La pobreza no viene por la disminución de las riquezas sino por la multiplicación de los deseos  No lastimes a los demás con lo que te causa dolor a ti mismo  Exígete mucho a ti mismo y espera poco de los demás Así te ahorrarás disgustos  El corazón tiene razones que la razón ignora  Pocas veces pensamos en lo que tenemos pero siempre en lo que nos falta  Nunca moriría por mis creencias porque podría estar equivocado  Nuestra vida siempre expresa el resultado de nuestros pensamientos dominantes  Si quieres encontrar los secretos del universo piensa en términos de energía frecuencia y vibración  Es un pequeño paso para un hombre pero un salto gigante para la humanidad  La aerodinámica es para fracasados que no saben hacer motores  Grandes descubrimientos y mejoras implican invariablemente la cooperación de muchas mentes  Los principios éticos elevados producen métodos comerciales eficaces  Si le hubiera preguntado a la gente qué querían habrían dicho caballos más rápidos  La persistencia es muy importante No debes renunciar al menos que te veas obligado a renunciar  El presente es de ellos el futuro para lo que realmente trabajé es mío  Si haces un producto superior la gente lo comprará  El primer paso es establecer que algo es posible entonces la probabilidad ocurrirá |

2.- crear corpus, stopwords, documentermmatrix, invertedmatriz, validate no zero

Se usó lenguaje R para el analizis de Kmeans.

Para la creación del corpus se usó el siguiente fragmento de código.

|  |
| --- |
| getwd()  setwd("C:/Users/ccordero/Desktop/Maestria/09\_MAIM/04\_mineria/")  # 1 we need to read the document  frases.fut.viajes <- read.csv('fraces\_cerveza\_filosofos\_inges.txt', stringsAsFactors = F, sep="|", header = F)  # frases.fut.viajes is a data frame  str(frases.fut.viajes)  class(frases.fut.viajes)  # frase returns a content with every line  class(frases.fut.viajes$frase)  frases.fut.viajes$frase[2]  colnames(frases.fut.viajes)  colnames(frases.fut.viajes) <- c("frase")  # 2 We need to convert the document in corpus to be able to work  datos.clase.junio.2020.df <- frases.fut.viajes  # we need to convert the data frame into a corpus to manipulate it as lines of text  base.corpus <- Corpus(VectorSource(datos.clase.junio.2020.df$frase)) |

Después de hacer el corpus se eliminaran los stopwords en español predeterminados por R y crear una matriz con la información se imprime la matriz y se analizan las palabras para revisar cuales no dan valor agregados. Del análisis yo concluí que las palabras siguientes no dan valor agregado: los, un, la.

|  |
| --- |
| base.corpous <- tm\_map(base.corpus, removeWords, c(stopwords("spanish"), "los", "un", "la")) |

Ya que se tiene la matriz filtrada con las palabras a usar se usa la funcio weightTfIdf para crear una matriz de pesos invertidos.

|  |
| --- |
| dtm <- weightTfIdf(dtm, normalize = TRUE) |

Del resultado se eliminan las frases que quedaron vacías ya sea por los stopwords o por el filtrado debido a la frecuencia de las palabras que se estableció en 1% y 80%.

|  |
| --- |
| dtm.matrix.tf.idf <- dtm.matrix.tf.idf[rowTotals > 0,] |

3.- determinar la cantidad de k

Para determinar k se usó el método grafico junto con determinación de codo, en la imagen siguiente se puede apreciar se hay tres codos en los puntos 2,7 y 13. El codo en 2 se ignorará ya que se tiene conocimiento de que son tres tipos de frases.

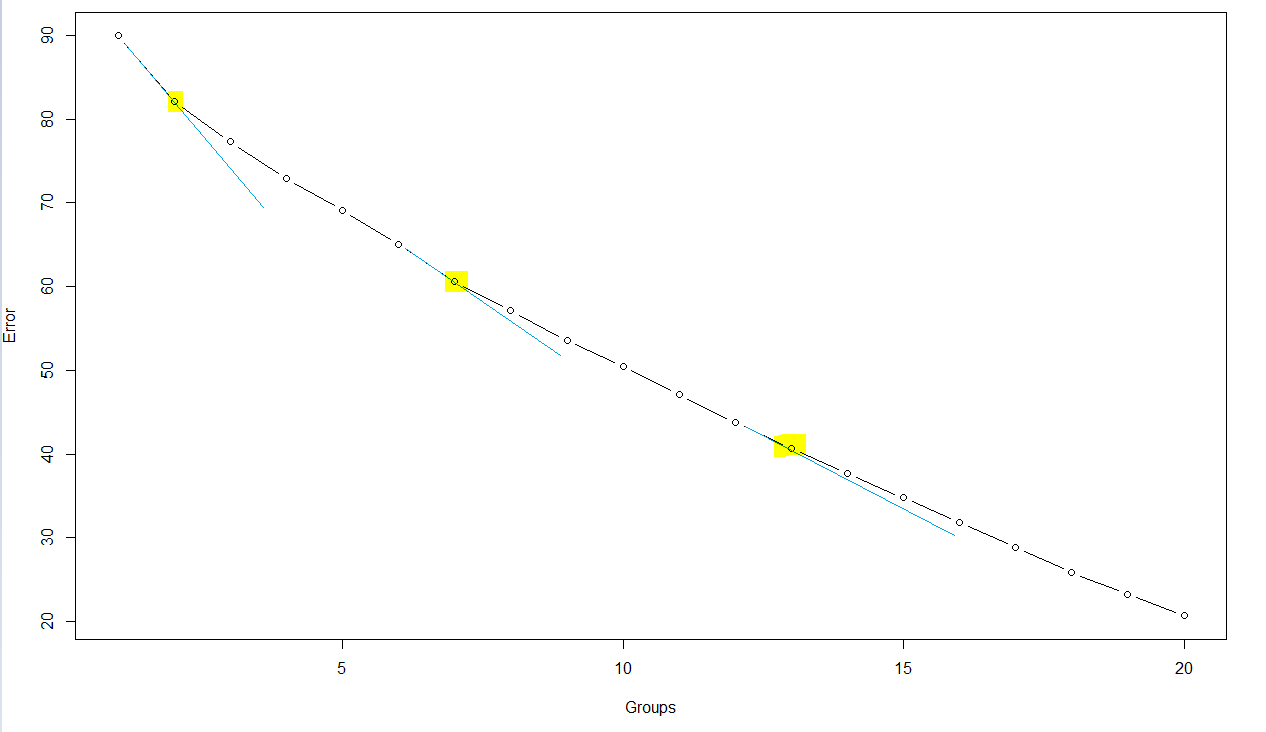


Figure Kmeans error vs k

La gráfica muestra el mean square error de la distancia de cada frase representada con un vector a su centroide.

4.- crear el modelo k-means

Ya que se tienen seleccionados los números de centroides analizar que serán 7 y 13 se ejecuta el siguiente código para generar el modelo.

|  |
| --- |
| set.seed(1000)  kmeans.model <- kmeans(base.kmeans, centers = 7, iter.max = 100)  # kmeans can also provide the cluster that every point belong with the atribute cluster  datos.clase.junio.2020.df$Grupo <- kmeans.model$cluster[1:dim(dtm.matrix.tf.idf)[1]]  # The result is that we have 2 goups and we can group them in this way :  # Results are much better if the lenght of the fraces is the same or similar  # 3 3 3 3 3 3 3 3 3 3 3 6 7 3 1 1 2 4 5 3 3 3 3 3 3 3 3 2 3 3  # Applying 7 centers  # 1,2,4,5,6,7 for filosofy  # it is not precise  set.seed(1000)  kmeans.model <- kmeans(base.kmeans, centers = 13, iter.max = 100)  # kmeans can also provide the cluster that every point belong with the atribute cluster  datos.clase.junio.2020.df$Grupo <- kmeans.model$cluster[1:dim(dtm.matrix.tf.idf)[1]]  # The result is that we have 2 goups and we can group them in this way :  # Results are much better if the lenght of the fraces is the same or similar  # 3 3 3 2 12 3 3 3 3 3 3 13 11 3 1 1 6 8 5 3 3 12 9 10 10 3 3 4 7 3  # Applying 13 centers  # 2,3,12 beer  # 1,5,6,8,11,13 for filosofy  # 4,7,9,10 for engineering  # it is a little bit more precise |

5.- discutir los resultados.

Si aplicamos solo 7 centroides los resultados son los siguientes.

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

3 3 3 3 3 3 3 3 3 3 3 6 7 3 1 1 2 4 5 3 3 3 3 3 3 3

27 28 29 30

3 2 3 3

Solo se puede distinguir que los centroides 1,2,4,5,6,7 son de filosofía, los de ingeniería y cerveza irónicamente están en el mismo centroide.

Usando 13 centrodes el resultado es el siguiente.

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 3 3 3 2 12 3 3 3 3 3 3 13 11 3 1 1 6 8 5 3 3 12 9 10 10 3

27 28 29 30

3 4 7 3

De ahí se puede ver que los centroides 2,3,12 son para la cerveza, 1,5,6,8,11,13 son para filosofía y 4,7,9,10 son para ingenieros. El resultado es mucho más preciso. Las frases que tienen errores serian: 11,14,20,21,22,26,27,30 lo que significa una precisión del 73%.

Para clasificar las frases de cerveza no se tiene ningún error, pero hay que tomar en consideración que todas las frases de cerveza incluían dicha palabra, para ingeniería y filosofía prácticamente todas la frases tenían palabras únicas lo que dificulta el análisis, quizá usando un dataset más amplio seria más preciso el algoritmo.

**Conclusiones**

\* Kmeans es un método de clustering que ayuda a clasificar distintos textos.

\* En Kmeans es importante usar el menor número de centroides posible para aligerar el procesamiento.

\* Mientras más centroides se usan mayor precisión se tendrá.

\* El clustering se hace más claro si se tienen las mismas palabras en distintas frases del mismo tipo, por lo tanto, es recomendable hacer uso de un dataset amplio.

**Código**

|  |
| --- |
| install.packages("tm")  library(tm)  #########################  getwd()  setwd("C:/Users/ccordero/Desktop/Maestria/09\_MAIM/04\_mineria/")  # 1 we need to read the document  frases.fut.viajes <- read.csv('fraces\_cerveza\_filosofos\_inges.txt', stringsAsFactors = F, sep="|", header = F)  # frases.fut.viajes is a data frame  str(frases.fut.viajes)  class(frases.fut.viajes)  # frase returns a content with every line  class(frases.fut.viajes$frase)  frases.fut.viajes$frase[2]  colnames(frases.fut.viajes)  colnames(frases.fut.viajes) <- c("frase")  # 2 We need to convert the document in corpus to be able to work  datos.clase.junio.2020.df <- frases.fut.viajes  # we need to convert the data frame into a corpus to manipulate it as lines of text  base.corpus <- Corpus(VectorSource(datos.clase.junio.2020.df$frase))  class(base.corpus)  ndocs <- length(base.corpus)  inspect(base.corpus)  # content returns a content with every line in a corpus  base.corpus$content[2]  # 3 we need to declare some limits and clean the text  # thresholds - ignore exremelly rare or common words  minTermFreq <- ndocs \* 0.01  maxTermFreq <- ndocs \* 0.8  stopwords("spanish")  stopwords("english")  # define stopwords that is a list of words that are going to be removed from the text  # by default the stopwords are english then we need to set the spanish in this case  # The third argument just have to be a list not necessarilly predefined stopwords, in fact  # it is much better to define or list of stop words  base.corpous <- tm\_map(base.corpus, removeWords, c(stopwords("spanish"), "los", "un", "la"))  # 4 Now we create a tokenized matrix representation of fraces vs word  # Example frace: Hello world world  # Hello world tv  # frace 1 2 0  dtm = DocumentTermMatrix(base.corpous,  control = list(  stopwords = TRUE,  wordLengths= c(1,Inf),  removePunctuation=T,  removeNumbers=T,  bounds = list(global = c(minTermFreq,maxTermFreq))  )  )  class(dtm)  # dtm will be an object we need to convert into a matrix  dtm.matrix <- as.matrix(dtm)  dtm.matrix  dim(dtm.matrix)  dtm.matrix[1:5,1:10]  # 5 We apply inverse documnet frequency  # First we need to understand that if a word appear just in few fraces it will be more usefull  # to classify. Then we are going to take the matrix that we have and divide all the numbers over  # the number of fraces that have it, and not only that also the number the times that appears  # in the frace and others.  # Remember that the method receives a DTM object not a matrix  # mathit{idf}\_i = \log\_2 \frac{|D|}{|\{d \mid t\_i \in d\}|  ?weightTfIdf  dtm <- weightTfIdf(dtm, normalize = TRUE)  dtm.matrix.tf.idf <- as.matrix(dtm)  nrow(dtm.matrix.tf.idf)  dtm.matrix.tf.idf[1:5,1:10]  class(dtm.matrix.tf.idf)  dim(dtm.matrix.tf.idf)  # remove empty rows because maybe after removing stopwords or too frequent or unfrequent words maybe a row is empty now  rowTotals <- apply(dtm.matrix.tf.idf,1,sum)  dtm.matrix.tf.idf <- dtm.matrix.tf.idf[rowTotals > 0,]  dim(dtm.matrix.tf.idf)  # 6 apply Kmeans  # Kmeans is designed to make clusters. We have to define a number of centroids, and every centroid, this start  # randomly positioned but they will be move to the center of the points closer to such centroid. Many iterations can be performed,  # or it can stops if the error is not changing.  # The error in Kmeans is the square mean error of the distances from the points of a cluster to its centroid. We have one error  # mesurement per cluster but we can add all of them to have a total.  # The most complicated point is to define the number of centroids.  # We are going to start doing an analysis to know how many centroids are ideal.  # we are going to apply kmeans from 1 centroid to 10 and check the error(withinss) then we are going to plot the result  # When we start watching progres we are going to take that result as the best number of centroids. We call it elbow rule.  base.kmeans <- dtm.matrix.tf.idf  # We need to treat it like dataframe  classified.docs <- as.data.frame(base.kmeans)  wss <- vector()  for(i in 1:20){  set.seed(1234)  wss[i] <- sum(kmeans(base.kmeans, centers = i, iter.max = 100)$withinss)  }  plot(1:20, wss, type="b", xlab="Groups", ylab="Error")  # We can see that after the point 2 and 4 there is a gradual improvement then we are going to take 2 centers because  # a rule in machine learning is to take the simples model as posible.  # 7 Evaluate the group  set.seed(1000)  kmeans.model <- kmeans(base.kmeans, centers = 2, iter.max = 100)  # kmeans can also provide the cluster that every point belong with the atribute cluster  datos.clase.junio.2020.df$Grupo <- kmeans.model$cluster[1:dim(dtm.matrix.tf.idf)[1]]  # The result is that we have 2 goups and we can group them in this way :  # 1 1 2 2 2 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  # 1 shall be for traveling and 2 for futbol  set.seed(1000)  kmeans.model <- kmeans(base.kmeans, centers = 7, iter.max = 100)  # kmeans can also provide the cluster that every point belong with the atribute cluster  datos.clase.junio.2020.df$Grupo <- kmeans.model$cluster[1:dim(dtm.matrix.tf.idf)[1]]  # The result is that we have 2 goups and we can group them in this way :  # Results are much better if the lenght of the fraces is the same or similar  # 3 3 3 3 3 3 3 3 3 3 3 6 7 3 1 1 2 4 5 3 3 3 3 3 3 3 3 2 3 3  # Applying 7 centers  # 1,2,4,5,6,7 for filosofy  # it is not precise  set.seed(1000)  kmeans.model <- kmeans(base.kmeans, centers = 13, iter.max = 100)  # kmeans can also provide the cluster that every point belong with the atribute cluster  datos.clase.junio.2020.df$Grupo <- kmeans.model$cluster[1:dim(dtm.matrix.tf.idf)[1]]  # The result is that we have 2 goups and we can group them in this way :  # Results are much better if the lenght of the fraces is the same or similar  # 3 3 3 2 12 3 3 3 3 3 3 13 11 3 1 1 6 8 5 3 3 12 9 10 10 3 3 4 7 3  # Applying 13 centers  # 2,3,12 beer  # 1,5,6,8,11,13 for filosofy  # 4,7,9,10 for engineering  # it is a little bit more precise |

**Referencias**

[1] « 25 FRASES INMORTALES SOBRE LA CERVEZA» [En línea]. Disponible en:

<https://www.thebeertimes.com/25-frases-inmortales-sobre-la-cerveza/> [Accedido: 22-mar-2020].

[2] « 75 frases filosóficas pronunciadas por grandes pensadores» [En línea]. Disponible en:

<https://psicologiaymente.com/reflexiones/frases-filosoficas> [Accedido: 22-mar-2020].

[3] « 57 Grandes Frases de Ingenieros Famosos» [En línea]. Disponible en:

<https://www.lifeder.com/frases-ingenieros-famosos/>