Para los ejemplos se utilizó el texto de una noticia del comercio que titula **solitario desenlace de un asesino, crónica sobre la muerte de Abimael Guzman**

1 PRE-PROCESAMIENTO

El pre-procesamiento de texto también ayuda a limpiar y estandarizar, lo que ayuda en sistemas analíticos, como aumentar la precisión de los clasificadores. También obtenemos adicional información y metadatos en forma de anotaciones, que también son muy útiles para dar más información sobre el texto.

Otro aspecto importante es comprender los datos textuales después del procesamiento y normalizándolo. Esto implicará volver a visitar algunos de los conceptos de sintaxis del lenguaje para comprender la estructura y la sintaxis del texto.

1.1 Tokenizacion

- Los tokens son componentes textuales independientes y mínimos que tienen una sintaxis definida y semántica. Un párrafo de texto o un documento de texto tiene varios componentes que incluyen oraciones que pueden desglosarse en frases y palabras.
- Las técnicas populares de tokenización incluyen la tokenización de oraciones y palabras, que son dividir un corpus de texto en oraciones y cada oración en palabras. Así, la tokenización se puede definir como el proceso de descomponer o dividir datos textuales en componentes significativos más pequeños llamados tokens.

1.1.1 Tokenización oraciones

- La tokenización de oraciones es el proceso de dividir un corpus de texto en oraciones que actúan como el primer nivel de fichas del que se compone el corpus. Esto también se conoce como segmentar el texto en oraciones significativas. Cualquier corpus es un cuerpo de texto donde cada párrafo comprende varias oraciones.
- Hay varias formas de realizar la tokenización de oraciones, son técnicas básicas que incluye buscar delimitadores específicos entre oraciones, como un punto (.) o un carácter de nueva línea (n) y, a veces, incluso un punto y coma.

En el siguiente ejemplo se puede apreciar que al aplicar la función **tokenize sentences** divide los parrafos en oraciones

```
library(tokenizers)
options(max.print = 25)

# Asignar espacio de trabajo
setwd(readClipboard())

# Cargar—Leer archivo de noticias.txt
noticias <- readLines("noticias.txt",encoding="UTF-8")

#tokenizacion por oraciones
tokenize_sentences(noticias)
```

```
[[1]]
[1] "A las 6:40 a.m. de este sábado 11, personal médico de la Base Naval del Callao constató que Abimael Guzmán Reinoso, cabecilla
 de Sendero Luminoso y el peor genocida en la historia de nuestro país, había dejado de existir."
[2] "Guzmán murió a los 86 años solo, envuelto en frazadas en la camilla del tópico de su centro de reclusión."

[3] "Al cierre de esta edición, la razón exacta de su muerte aún se desconocía, debido a que se continuaba realizando la necropsia
 correspondiente."
[4] "El INPE, en un breve comunicado, dijo que su fallecimiento se debió a complicaciones en su estado de salud."
```

1.1.2 Tokenización palabras

La tokenización de palabras es el proceso de dividir o segmentar oraciones en sus palabras, una oración es una colección de palabras, y con la tokenización esencialmente se divide una oración en una lista de palabras que se pueden usar para reconstruir la frase.

La tokenización de palabras es muy importante en muchos procesos, especialmente en la limpieza. normalizando el texto donde las operaciones como la derivación y la lematización funcionan como cada palabra individual basada en sus respectivos tallos y lema.

En el siguiente ejemplo se puede apreciar que al aplicar la función **tokenize words** divide los parrafos en oraciones y estos a su vez en cada palabra que compone la oración

```
#tokenizacion por palabras
tokenize_words(noticias)
```

```
[[1]]
 [1] "a"         "las"       "6"         "40"        "a.m"       "de"        "este"      "sábado"    "11"        "personal"
[11] "médico"    "de"        "la"        "base"      "naval"     "del"       "callao"    "constató"  "que"       "abimael"
[21] "guzmán"    "reinoso"   "cabecilla" "de"        "sendero"
 [ reached getOption("max.print") -- omitted 77 entries ]

[[2]]
 [1] "guzmán"      "murió"       "mientras"    "cumplía"     "dos"         "sentencias"  "de"          "cadena"
 [9] "perpetua"    "por"         "las"         "atrocidades" "que"         "cometió"     "junto"       "a"
[17] "su"          "horda"       "de"          "fanáticos"   "cuyos"       "atentados"   "comenzaron"  "el"
[25] "17"
 [ reached getOption("max.print") -- omitted 76 entries ]
```

1.2 Normalización

La normalización de texto se define como un proceso que consiste en una serie de pasos que debe seguirse para limpiar y estandarizar datos textuales en un formulario que podría ser consumido por otros sistemas y aplicaciones de NLP, etc

A menudo, la tokenización en sí misma también es parte de la normalización del texto. Además de la tokenización, varias otras técnicas incluyen limpieza de texto, conversión de mayúsculas, corrección ortográfica, eliminando palabras vacías y otros términos innecesarios, derivación y lematización. Texto

1.2.1 Limpieza Caracteres

Una tarea importante en la normalización de texto consiste en eliminar innecesarios y especiales, estos pueden ser símbolos especiales o incluso signos de puntuación que ocurren en las oraciones.

Este paso a menudo se realiza antes o después de la tokenización. La razón principal para hacerlo es porque a menudo la puntuación o los caracteres especiales no tienen mucha importancia cuando se analiza el texto y se usa para extraer funciones o información basada en NLP y ML.

En el siguiente ejemplo se realiza la limpieza del texto, quitando los espacios en blanco, signos de puntuación y datos numéricos, para esto se hace uso de la librería **"tm"**

```
#limpieza
library(tm)
corpus <- VCorpus(VectorSource(noticias))
d  <- tm_map(corpus, content_transformer(tolower))
d  <- tm_map(d, stripWhitespace)
d <- tm_map(d, removePunctuation)
d <- tm_map(d,removeNumbers)
d
d[["1"]][["content"]]

######forma 2###########
library(tidyverse)
library(dplyr)

limpiar_tokenizar <- function(texto){
  # El orden de la limpieza no es arbitrario
  # Se convierte todo el texto a min sculas
  nuevo_texto <- tolower(texto)
  # Eliminaci n de p ginas web (palabras que empiezan por "http." seguidas
  # de cualquier cosa que no sea un espacio)
  nuevo_texto <- str_replace_all(nuevo_texto,"http\\S*", "")
  # Eliminaci n de signos de puntuaci n
  nuevo_texto <- str_replace_all(nuevo_texto,"[[:punct:]]", " ")
  # Eliminaci n de n meros
  nuevo_texto <- str_replace_all(nuevo_texto,"[[:digit:]]", " ")
  # Eliminaci n de espacios en blanco m ltiples
  nuevo_texto <- str_replace_all(nuevo_texto,"[\\s]+", " ")
  # Eliminacion de tildes
  nuevo_texto <- chartr('            ','aeioun',nuevo_texto)
  # Tokenizaci n por palabras individuales
  nuevo_texto <- str_split(nuevo_texto, " ")[[1]]
  # Eliminaci n de tokens con una longitud < 2
  nuevo_texto <- keep(.x = nuevo_texto, .p = function(x){str_length(x) > 1})
```

```
  return(nuevo_texto)
}

df.noticias=as.data.frame(noticias)
df.noticias <- df.noticias %>% mutate(texto_tokenizado = map(.x = df.noticias,
                                                             .f = limpiar_tokenizar))

df.noticias %>% select(texto_tokenizado) %>% head()
df.noticias %>% slice(1) %>% select(texto_tokenizado) %>% pull()
df.noticias$ID <- seq.int(nrow(df.noticias))
str(df.noticias)
```

```
> d
<<VCorpus>>
Metadata:  corpus specific: 0, document level (indexed): 0
Content:  documents: 11
> d[["1"]][["content"]]
[1] "a las  am de este sábado  personal médico de la base naval del callao constató que abimael guzmán
reinoso cabecilla de sendero luminoso y el peor genocida en la historia de nuestro país había dejado de
 existir guzmán murió a los  años solo envuelto en frazadas en la camilla del tópico de su centro de re
clusión al cierre de esta edición la razón exacta de su muerte aún se desconocía debido a que se contin
uaba realizando la necropsia correspondiente el inpe en un breve comunicado dijo que su fallecimiento s
e debió a complicaciones en su estado de salud"
>
```

1.2.2 Limpieza Stopwords

- Las palabras vacías, son palabras que tienen poca o ninguna importancia, por lo general, se eliminan del texto durante el procesamiento para retener las palabras que tienen una significación y contexto.
- Las palabras vacías generalmente son palabras que terminan ocurriendo más si agregaste cualquier corpus de texto basado en tokens singulares y verificaras su frecuencias, palabras como los artículos son palabras vacías.
- No hay una lista universal o una lista exhaustiva de palabras vacías, cada dominio o idioma puede tener su propio conjunto de palabras.

En el siguiente ejemplo se realiza la limpieza de los stopwords tanto desde las palabras incluidas en la librería y desde una base personalizada

```
#limpieza
library(tm)
corpus <- VCorpus(VectorSource(noticias))
d  <- tm_map(corpus, content_transformer(tolower))
d  <- tm_map(d, stripWhitespace)
d <- tm_map(d, removePunctuation)
d <- tm_map(d,removeNumbers)
d
d[["1"]][["content"]]

######forma 2###########
library(tidyverse)
library(dplyr)

limpiar_tokenizar <- function(texto){
  # El orden de la limpieza no es arbitrario
  # Se convierte todo el texto a min sculas
  nuevo_texto <- tolower(texto)
  # Eliminaci n de p ginas web (palabras que empiezan por "http." seguidas
  # de cualquier cosa que no sea un espacio)
  nuevo_texto <- str_replace_all(nuevo_texto,"http\\S*", "")
  # Eliminaci n de signos de puntuaci n
  nuevo_texto <- str_replace_all(nuevo_texto,"[[:punct:]]", " ")
  # Eliminaci n de n meros
  nuevo_texto <- str_replace_all(nuevo_texto,"[[:digit:]]", " ")
  # Eliminaci n de espacios en blanco m ltiples
  nuevo_texto <- str_replace_all(nuevo_texto,"[\\s]+", " ")
  # Eliminacion de tildes
  nuevo_texto <- chartr('            ','aeioun',nuevo_texto)
  # Tokenizaci n por palabras individuales
  nuevo_texto <- str_split(nuevo_texto, " ")[[1]]
  # Eliminaci n de tokens con una longitud < 2
  nuevo_texto <- keep(.x = nuevo_texto, .p = function(x){str_length(x) > 1})
```

```
  return(nuevo_texto)
}

df.noticias=as.data.frame(noticias)
df.noticias <- df.noticias %>% mutate(texto_tokenizado = map(.x = df.noticias,
                                                             .f = limpiar_tokenizar))

df.noticias %>% select(texto_tokenizado) %>% head()
df.noticias %>% slice(1) %>% select(texto_tokenizado) %>% pull()
df.noticias$ID <- seq.int(nrow(df.noticias))
str(df.noticias)

# Limpieza- Stop Words - Spanish
#palabras por defecto de la libreria
d <- tm_map(d, removeWords, stopwords("spanish"))
d[["1"]][["content"]]

####lista personalizada 1###############
sw <- readLines("stopwordses.txt",encoding="UTF-8")
sw = iconv(sw, to="ASCII//TRANSLIT")
d <- tm_map(d, removeWords, sw)
d[["1"]][["content"]]


####lista personalizada 2
noticias.raw <- df.noticias %>% select(-noticias) %>% unnest()
head(noticias.raw)

# Se filtran las stopwords
noticias_tidy <- noticias.raw %>% filter(!(texto_tokenizado %in% sw))
head(noticias_tidy)
```

```
> d[["1"]][["content"]]
[1] "  am  sábado personal médico  base naval  callao constató  abimael guzmán reinoso cabecilla  s
endero luminoso  peor genocida  historia  país  dejado  existir guzmán murió  años solo envuelto
frazadas  camilla  tópico  centro  reclusión  cierre  edición  razón exacta  muerte aún  desconocía
 debido  continuaba realizando  necropsia correspondiente  inpe  breve comunicado dijo  fallecimien
to  debió  complicaciones  salud"
```

1.3 stemming

El morfema es la unidad independiente más pequeña en cualquier lenguaje natural, consisten en unidades que son tallos y afijos.

Los afijos son unidades como prefijos, sufijos, etc., que están unidos a una raíz de palabra para cambiar su significado o crear una nueva palabra por completo, los tallos de palabras también se conocen a menudo como la forma básica de una palabra, y podemos crear nuevas palabras adjuntando afijos en un proceso conocido como Inflexión.

El reverso de esto es obtener la forma base de una palabra de su forma flexionada y esto se conoce como derivación (stemming).

1.4 lemmatization

La lematización reduce las palabras inflexionadas de manera adecuada, asegurando que la palabra raíz pertenezca al idioma. Toma en consideración el análisis morfológico de las palabras.

Para hacerlo, es necesario tener un diccionario detallado para que el algoritmo pueda revisar para vincular el formulario a su lema.

En Lematización la palabra raíz se llama Lema, un lema es la forma canónica, forma de diccionario o forma de cita de un conjunto de palabras, la clave de esta metodología es la lingüística.

```
#######stemming#################
library(pacman)
library(textstem)
dw <- c('terrorismo', 'terrorista')
stem_words(dw)

stem_strings(noticias)

#######lematization#################
dw <- c('terrorismo', 'terrorista')
lemmatize_words(dw)
```

```
lemmatize_strings(noticias)

####lemtizacion_espa ol#########
##############stemming y lemmantizacion#########
library(SnowballC)
library(corpus)

tabes=read.delim("lemmatization-es.txt",header = FALSE, sep = "",
                 stringsAsFactors = FALSE)
names(tabes) <- c("stem", "term")
head(tabes,10)

stem_liste <- function(term) {
  i <- match(term, tabes$term)
  if (is.na(i)) {
    stem <- term
  } else {
    stem <- tabes$stem[[i]]
  }
  stem
}

# text_tokens(noticias, stemmer = stem_liste)

names(noticias_tidy)[1] <- "term"
noticias_tidy=noticias_tidy %>%left_join(tabes, by = "term")
noticias_tidy$lemma=ifelse(is.na(noticias_tidy$stem ), noticias_tidy$term, noticias_tidy$
    stem)


noticias_tidy %>% group_by(term) %>% count(term)%>%
  arrange(desc(n))
```

codigo_parte1.R | noticias_tidy | tokenizar.R

Filter

	term	ID	stem	lemma
100	comunicado	1	comunicar	comunicar
101	comunicado	2	comunicar	comunicar
102	comunicado	3	comunicar	comunicar
103	comunicado	4	comunicar	comunicar
104	comunicado	5	comunicar	comunicar
105	comunicado	6	comunicar	comunicar
106	comunicado	7	comunicar	comunicar
107	comunicado	8	comunicar	comunicar
108	comunicado	9	comunicar	comunicar
109	comunicado	10	comunicar	comunicar
110	comunicado	11	comunicar	comunicar
111	constato	1	constatar	constatar
112	constato	2	constatar	constatar
113	constato	3	constatar	constatar
114	constato	4	constatar	constatar
115	constato	5	constatar	constatar

Gráfico donde se muestra la frecuencia de las palabras de la noticia sin realizar ningún tipo de limpieza

```
noticias.raw %>% group_by(texto_tokenizado) %>% count(texto_tokenizado)%>%
  filter(n >= 10) %>%arrange(desc(n)) %>%
  ggplot(aes(x = reorder(texto_tokenizado,n), y = n)) +
  ggtitle("frecuencia de palabras sin realizar la limpieza") +
  theme(axis.text.x = element_text(angle=90,hjust=1))+
  geom_col(fill="blue") +
  #theme_bw() +
  labs(y = "", x = "")
```

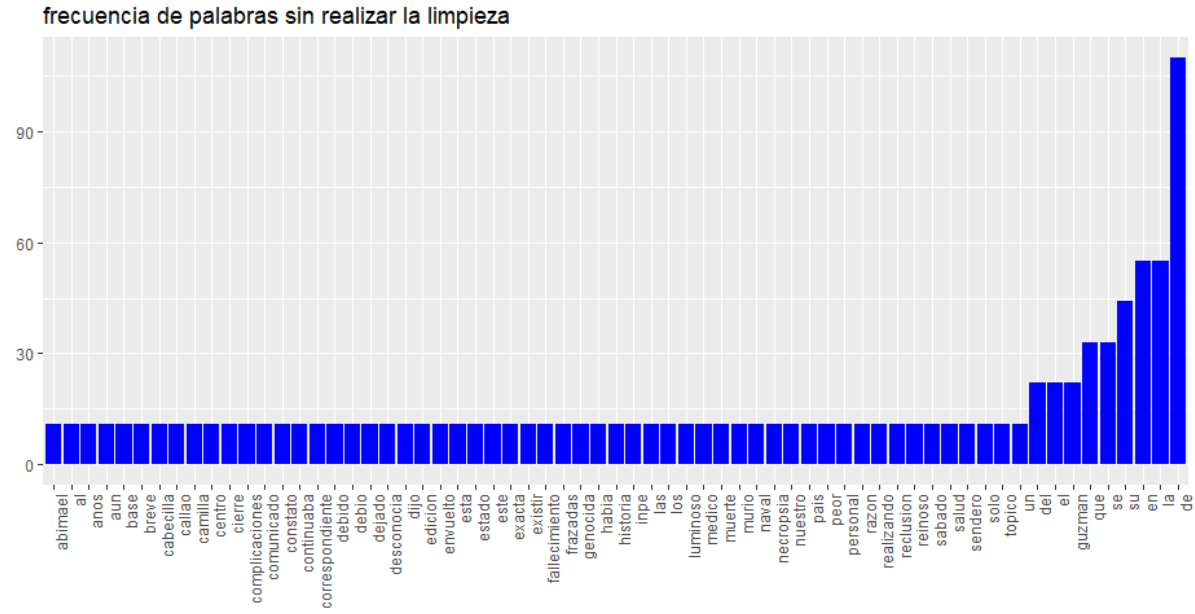
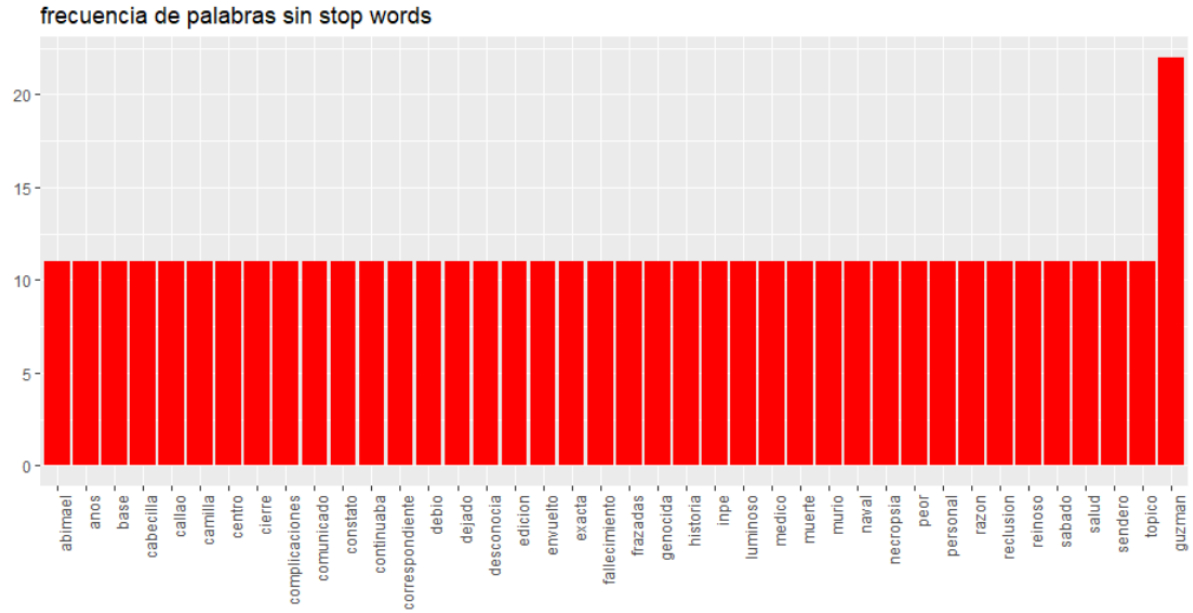


Gráfico donde se muestra la frecuencia de las palabras de la noticia después de realizar la limpieza de los stopwords

```
noticias_tidy %>% group_by(term) %>% filter(!(term %in% sw))%>%
  count(term)%>% filter(n >= 5) %>%arrange(desc(n))%>%
  ggplot(aes(x = reorder(term,n), y = n)) +
  ggtitle("frecuencia de palabras sin stop words")+
  geom_col(fill="red") +
  #theme_bw() +
  theme(axis.text.x = element_text(angle=90,hjust=1))+
  labs(y = "", x = "")
```



1.5 Extracción de Variables

1.5.1 TDM

```
d_tdm <- TermDocumentMatrix(d)
d_tdm
m <- as.matrix(d_tdm)
dim(m)
v <- sort(rowSums(m),decreasing=TRUE)
```

```
df <- data.frame(word = names(v),freq=v)
```

```
> d_tdm
<<TermDocumentMatrix (terms: 269, documents: 11)>>
Non-/sparse entries: 325/2634
Sparsity           : 89%
Maximal term length: 15
Weighting          : term frequency (tf)
```

1.5.2 DTM

```
#####DTM############
d_dtm <- DocumentTermMatrix(d)
d_dtm
m2 <- as.matrix(d_dtm)
```

```
> d_dtm
<<DocumentTermMatrix (documents: 11, terms: 269)>>
Non-/sparse entries: 325/2634
Sparsity           : 89%
Maximal term length: 15
Weighting          : term frequency (tf)
```

Anexo

Texto utilizado en los ejemplos

A las 6:40 a.m. de este sábado 11, personal médico de la Base Naval del Callao constató que Abimael Guzmán Reinoso, cabecilla de Sendero Luminoso y el peor genocida en la historia de nuestro país, había dejado de existir. Guzmán murió a los 86 años solo, envuelto en frazadas en la camilla del tópico de su centro de reclusión. Al cierre de esta edición, la razón exacta de su muerte aún se desconocía, debido a que se continuaba realizando la necropsia correspondiente. El INPE, en un breve comunicado, dijo que su fallecimiento se debió a complicaciones en su estado de salud.

Guzmán murió mientras cumplía dos sentencias de cadena perpetua por las atrocidades que cometió junto a su horda de fanáticos, cuyos atentados comenzaron el 17 de mayo de 1980 en Chuschi (Ayacucho) y continuaron hasta poco antes de su captura, el 12 de setiembre de 1992 en Lima. El costo de la insania homicida del cabecilla de Sendero fue la pérdida de las vidas de más de 32 mil peruanos, entre militares, policías, funcionarios, madres, padres de familia y menores de edad, sobre todo de las zonas rurales y más pobres del país, según la Comisión de la Verdad y Reconciliación.

Lo último que se sabía sobre la salud del terrorista es que el pasado 5 de agosto había sido dado de alta del hospital de la Base Naval luego de haber estado más de dos semanas internado por una infección, por lo que se le tuvo que inyectar medicamentos vía intravenosa. Una atención que se rehusaba a recibir, ya que prefería quedarse en su celda, pero finalmente se concretó el 20 de julio. Según la jefa del INPE, Susana Silva, hasta el viernes Guzmán fue atendido por un médico geriatra del Ministerio de Salud al presentar un decaimiento extremo e inapetencia. De hecho, se tenía previsto que una junta médica lo evaluara el sábado por la mañana. Pero a las 7 a.m. de ayer Silva recibió la llamada del jefe de la Base Naval en la cual le informaba que ya no era necesario: Guzmán había muerto un día antes de cumplirse 29 años de su captura.

Guzmán siendo examinado en su celda el pasado 19 de julio. Se oponía a recibir atención médica, algo que finalmente pasó al día siguiente. Poco después de las 8 a.m., el fiscal provincial Julio García Odón se presentó en el centro penitenciario para levantar el cadáver. El Ministerio Público informó que, además de la necropsia de ley, al cuerpo de Guzmán se le harían exámenes antropológicos, de odontología forense, biológicos y de ADN en la Morgue Central del Callao.

El ministro de Justicia, Aníbal Torres, dijo que, a su criterio, lo más conveniente sería que los restos del terrorista sean incinerados para que sus simpatizantes no tengan un lugar donde rendirle homenaje. Pero ello, afirmó, es decisión de la fiscalía, así como la presentación del cuerpo en la morgue a su esposa, la número dos de Sendero, Elena Yparraguirre (interna en el penal Virgen de Fátima).

Torres también recalcó que enaltecer, hacer movilizaciones en memoria de Guzmán o realizar propaganda a su favor califica como delito de apología del terrorismo, cuyas penas van de los 4 a 15 años de cárcel.

Si Guzmán terminó su vida en la cárcel, se debe a que fue procesado y juzgado por la justicia civil de un Estado de derecho que quiso derrumbar. Luz Ibáñez, hoy jueza de la Corte Penal Internacional, fue la fiscal que logró la condena de cadena perpetua en el 2006 (en el 2003 quedó anulada la sentencia que dictó un fuero de jueces sin rostro en octubre de 1992).

Desde La Haya, Ibáñez le dijo a El Comercio: “Murió la persona, pero eso no significa que haya pasado lo mismo con la ideología violentista y de terror que sedujo y llevó a miles de peruanos, sobre todo jóvenes, a pensar que esos caminos eran agentes de cambio. Este debe ser un momento de unidad entre peruanos y pensar que las estructuras sociales de desigualdad e injusticia perviven en el Perú con los más excluidos. Eso sigue siendo un caldo cultivo”.

El exministro del Interior Carlos Morán fue integrante, como capitán de la PNP, del equipo del GEIN que consiguió el arresto de Guzmán. Ante la muerte del terrorista, expresó: “Ni alegría ni tristeza. Solo pensar que no nos equivocamos cuando lo capturamos. A pesar de ser responsable de miles de muertes, se le respetó su integridad física para que fuera procesado por la justicia peruana”.

Guzmán muere con juicios pendientes. Dos de ellos: la matanza de 117 campesinos en Soras y el Caso Perseo, relacionado al último vestigio que quedó de esa ideología fanática: el Movadef.