

# I Have a Dream Wordcloud

We'll need a few libraries to get started: one for mining text and one for creating wordclouds. Then, we will read the speech in line-by-line, a departure from typical R input in dataframes.

```
library(wordcloud)    # wordcloud generator
```

```
## Loading required package: RColorBrewer
```

```
library(tm)           # text-mining library
```

```
## Loading required package: NLP
```

```
# read in our data
```

```
r.text <- readLines("ihad.txt")
```

A corpus is an organized library of words taken from the raw text of the input data.

```
#create a corpus object from the text
```

```
r.corpus <- VCorpus(VectorSource(r.text))
```

The text mining library provides facilities to “clean” our data. Using the `tm_map()` function, we can remove punctuation, whitespace, and numbers to make our corpus tidy.

Crucially, we can also remove “stopwords”, words that serve a rudimentary function but convey little meaning on their own. They make for a pretty boring wordcloud, and don't let us see and context or theme. There is no defined list of them in English, but they are words like like “and”, “with”, “to”, etc. Below we call the function to create ‘c.corpus’, for clean corpus, and then update it step by step to remove things we don't want. Typically, at this stage we might also opt to try to “stem” the words, that is, use a library to turn them into their root form. Discovering, discovery, and discovered would all become “discover”. These programs often run into trouble with “er” endings and the letter “y”, turning “hundred” into “hundr” and “satisfy” into “satisfi”, etc. I opted to see if how this program would work without it, and, other than the somewhat redundant “America” and “American”, it is better unstemmed.

```
# clean the words: whitespace, punctuation, remove numbers, stem, stopwords
```

```
c.corpus <- tm_map(r.corpus, content_transformer(removePunctuation))
```

```
c.corpus <- tm_map(c.corpus, content_transformer(removeNumbers))
```

```
c.corpus <- tm_map(c.corpus, content_transformer(stemDocument))
```

```
c.corpus <- tm_map(c.corpus, content_transformer(stripWhitespace))
```

```
c.corpus <- tm_map(c.corpus, removeWords, stopwords())
```

Now, we pass the corpus to the `wordcloud()` function with a few optional arguments. We control if the order of appearance is random or not

```
# set a palate
```

```
pal <-brewer.pal(8,"Dark2")
```

```
# create a wordcloud
```

```
wordcloud(c.corpus, min.freq = 2, random.order = FALSE, colors = pal)
```

```
## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
```

```
## 'god' in 'mbsToSbcs': dot substituted for <e2>
```

```
## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
```

```
## 'god' in 'mbsToSbcs': dot substituted for <80>
```

```
## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
```

```
## 'god' in 'mbsToSbcs': dot substituted for <99>
```

```
## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
```

```

## rotWord * : conversion failure on 'god' in 'mbcsToSbcs': dot substituted
## for <e2>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on 'god' in 'mbcsToSbcs': dot substituted
## for <80>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on 'god' in 'mbcsToSbcs': dot substituted
## for <99>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : font metrics unknown for Unicode character U+2019

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## 'negro' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## 'negro' in 'mbcsToSbcs': dot substituted for <80>

## Warning in strwidth(words[i], cex = size[i], ...): conversion failure on
## 'negro' in 'mbcsToSbcs': dot substituted for <99>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on 'negro' in 'mbcsToSbcs': dot substituted
## for <e2>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on 'negro' in 'mbcsToSbcs': dot substituted
## for <80>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : conversion failure on 'negro' in 'mbcsToSbcs': dot substituted
## for <99>

## Warning in text.default(x1, y1, words[i], cex = size[i], offset = 0, srt =
## rotWord * : font metrics unknown for Unicode character U+2019

```

