# Sentimental analysis of top 100 Billboard charts

*Ingrid Brizotti*

Accordingly to Wikipedia, the Billboard Hot 100 is the music industry standard record chart in the United States for singles, published weekly by Billboard magazine. Chart rankings are based on sales, radio play, and online streaming

The goal is to analyze 30 years of lyrics using top 100 from Billboard. I used web scraping code from Kaylin Walker, and I add one more year 2016. The dataset is available in my folder and contains data from 1980 to 2016.

For the sentimenal analysis part I used the amazing package tidytext created by Julia Silge and David Robinson. Also Julia teaches a great course about this in DataCamp, you should definitely check it out ;)

Packages:

```
library(SnowballC)
library(tm)
library(stringr)
library(ggplot2)
library(RColorBrewer)
library(wordcloud)
library(tidytext)
library(tidyr)
library(dplyr)
```

Load the data and check the structure

```
load("~/Documents/GitHub/sentimental analysis billboard/billboard_1980_2016.Rda")
head(billboard_1980_2016)
```

```
## # A tibble: 6 x 6
##    Rank                              Song            Artist  Year
##   <chr>                             <chr>             <chr> <chr>
## 1     1                           call me           blondie  1980
## 2     2 another brick in the wall part ii        pink floyd  1980
## 3     3                             magic olivia newtonjohn  1980
## 4     4                     rock with you   michael jackson  1980
## 5     5       do that to me one more time captain  tennille  1980
## 6     6    crazy little thing called love             queen  1980
## # ... with 2 more variables: Lyrics <chr>, Source <chr>
```

Let's create a word cloud for the name of the songs, and the steps are:

1) create a corpus
2) convert the corpus to a plain text document
3) remove all punctuation and stopwords (example: I, me, my, and, etc)
4) stemming (example: learning -> learn, walked -> walk. . . )

5) plot the world cloud

```
bl <- billboard_1980_2016
song_corpus <- Corpus(VectorSource(bl$Song))
song_corpus <- tm_map(song_corpus, PlainTextDocument)
song_corpus <- tm_map(song_corpus, removePunctuation)
song_corpus <- tm_map(song_corpus, removeWords, stopwords('english'))
song_corpus <- tm_map(song_corpus, stemDocument)
song_corpus <- Corpus(VectorSource(song_corpus))

set.seed(3435)
wordcloud(words = song_corpus,max.words=100,
          random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
```


On data preparation, let's transform some variables to numeric, and transform the Lyrics column to a word column (example: unnest_tokens(name_output_column,input_column))

```
bl$Rank <- as.numeric(bl$Rank)
bl$Year <- as.numeric(bl$Year)

tidy_lyrics <- bl %>% unnest_tokens(word,Lyrics)
```

Check the frequency of words

```
tidy_lyrics %>% count(word,sort=TRUE)
```

```
## # A tibble: 37,092 x 2
##     word     n
##    <chr> <int>
##  1   you 53893
##  2     i 47876
##  3   the 42832
##  4    to 28798
##  5    me 25950
##  6   and 25806
```

```
##  7     a 23292
##  8    it 21920
##  9    my 18799
## 10    in 14877
## # ... with 37,082 more rows
```

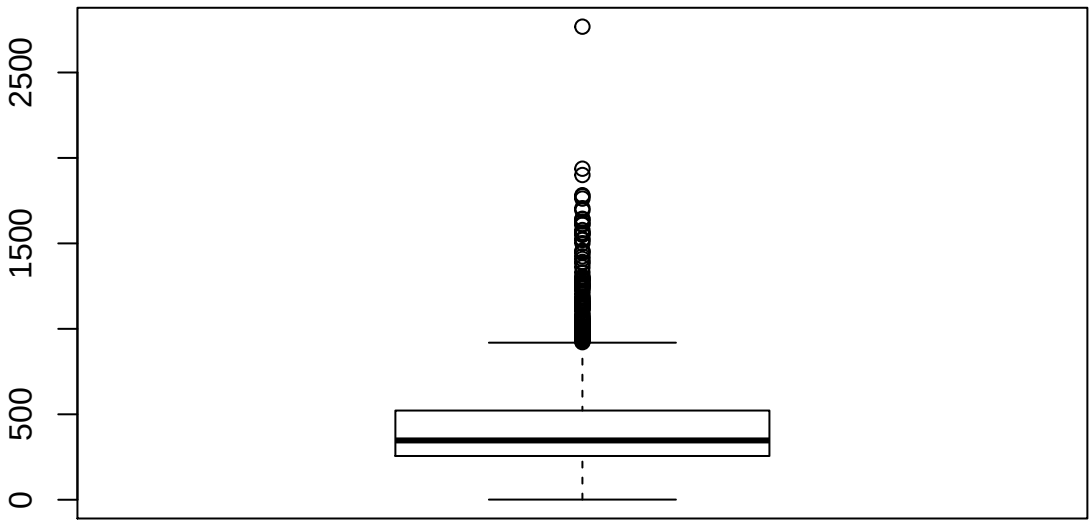
Choose top 200 words to do a word cloud

```
top200 <- tidy_lyrics %>% count(word,sort=TRUE)
top200 <- top200[c(1:200),]
wordcloud(words = top200$word, freq=top200$n,
          random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
```


Calculate total words per song and plot

```
total <- tidy_lyrics %>%
         count(Song) %>%
         rename(total_words = n)

boxplot(total$total_words)
```




```
summary(total$total_words)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0   256.0   347.0   414.9   522.0  2768.0
```

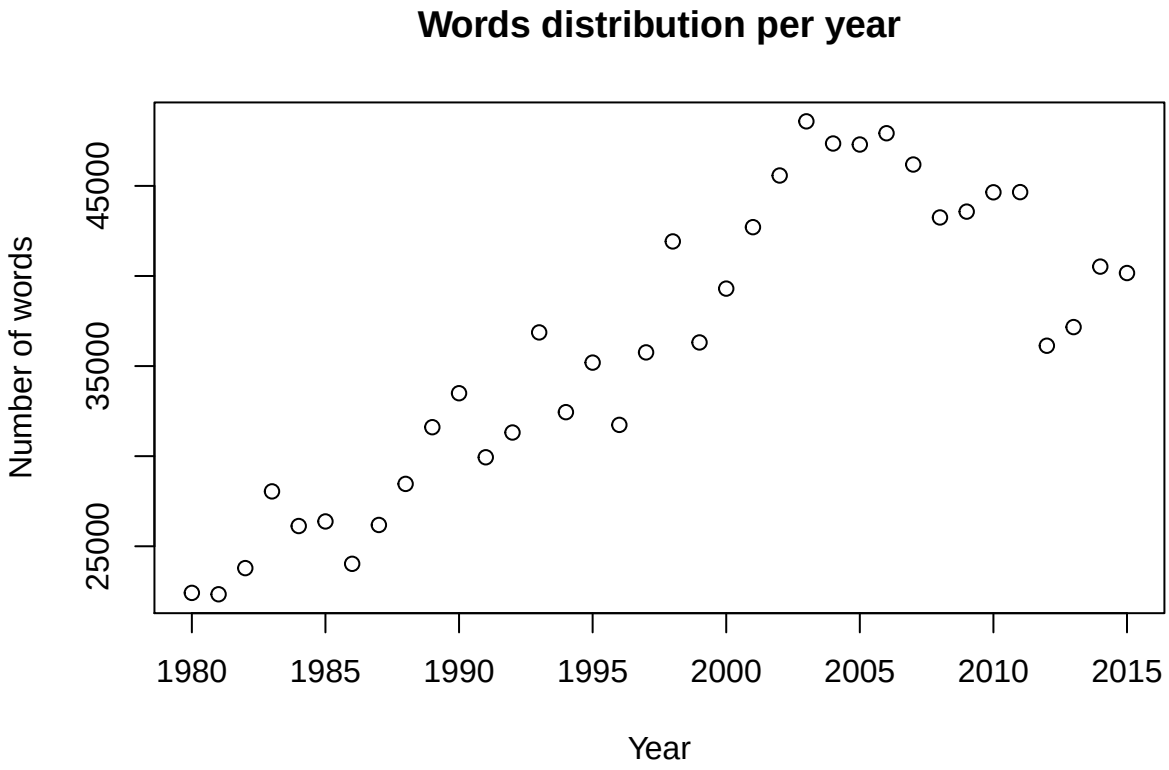
On average a song has 415 words

Combine total with tidy lyrics

```
lyric_count <- tidy_lyrics %>% left_join(total, by="Song")
```

Now let's check if the number of words per lyric increased during these 30 years

```
tot_year <- tidy_lyrics %>% count(Year) %>%
  rename(total_words = n)
# Since 2016 has many missing in the lyrics field, let's exclude this year
tot_year <- tot_year[1:36,]
plot(tot_year$Year, tot_year$total_words,
     xlab="Year", ylab="Number of words", main="Words distribution per year")
```




The number of words increased since 1980

**Sentimental Analysis**

Implement sentimental analysis using NRC lexicon (has 10 categories of sentiment: anger, anticipation, disgust, fear, joy, negative, positive, sadness, surprise and trust) More details check http://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm

```
lyric_sentiment <- lyric_count %>% inner_join(get_sentiments('nrc'))
```

```
## Joining, by = "word"
```

take a look at the lyric_sentiment and you'll see most of words have more than one sentiment

```
head(lyric_sentiment)
```

```
## # A tibble: 6 x 8
##    Rank    Song  Artist  Year Source    word total_words sentiment
##   <dbl>   <chr>   <chr> <dbl>  <chr>   <chr>       <int>     <chr>
## 1     1 call me blondie  1980      1    baby         236       joy
## 2     1 call me blondie  1980      1    baby         236  positive
## 3     1 call me blondie  1980      1 darling         236       joy
## 4     1 call me blondie  1980      1 darling         236  positive
## 5     1 call me blondie  1980      1 darling         236     trust
## 6     1 call me blondie  1980      1   chart         236     trust
```

Find how many sentiment each song has

```
lyric_sentiment %>% count(Song, sentiment, sort=TRUE)
```

```
## # A tibble: 29,012 x 3
##              Song sentiment     n
##             <chr>     <chr> <int>
##  1           baby  positive   231
##  2           baby       joy   226
##  3      real love  positive   213
##  4          angel  positive   193
##  5      disturbia  negative   182
##  6 live your life  positive   174
##  7          angel       joy   164
##  8           damn  negative   164
##  9      disturbia   sadness   164
## 10      disturbia   disgust   160
## # ... with 29,002 more rows
```

What songs have the highest proportion of joy words?

```
lyric_sentiment %>% count(Song, sentiment, total_words) %>%     #count using three arguments
                    ungroup() %>%
                    mutate(percent = n/total_words) %>%         # make a percent column
                    filter(sentiment=="positive") %>%
                    arrange(desc(percent))
```

```
## # A tibble: 3,097 x 5
##                       Song sentiment total_words     n   percent
##                      <chr>     <chr>       <int> <int>     <dbl>
##  1                  axel f  positive           1     1 1.0000000
##  2                children  positive           1     1 1.0000000
##  3       hooked on classics  positive           1     1 1.0000000
##  4         miami vice theme  positive           1     1 1.0000000
##  5          sadeness part i  positive           1     1 1.0000000
##  6 keep feeling fascination  positive         189    43 0.2275132
##  7               lucky star  positive         352    80 0.2272727
##  8                true blue  positive         445    97 0.2179775
##  9  lean wit it rock wit it  positive         670   146 0.2179104
## 10       you give good love  positive         381    82 0.2152231
## # ... with 3,087 more rows
```

And the proportion for sad words

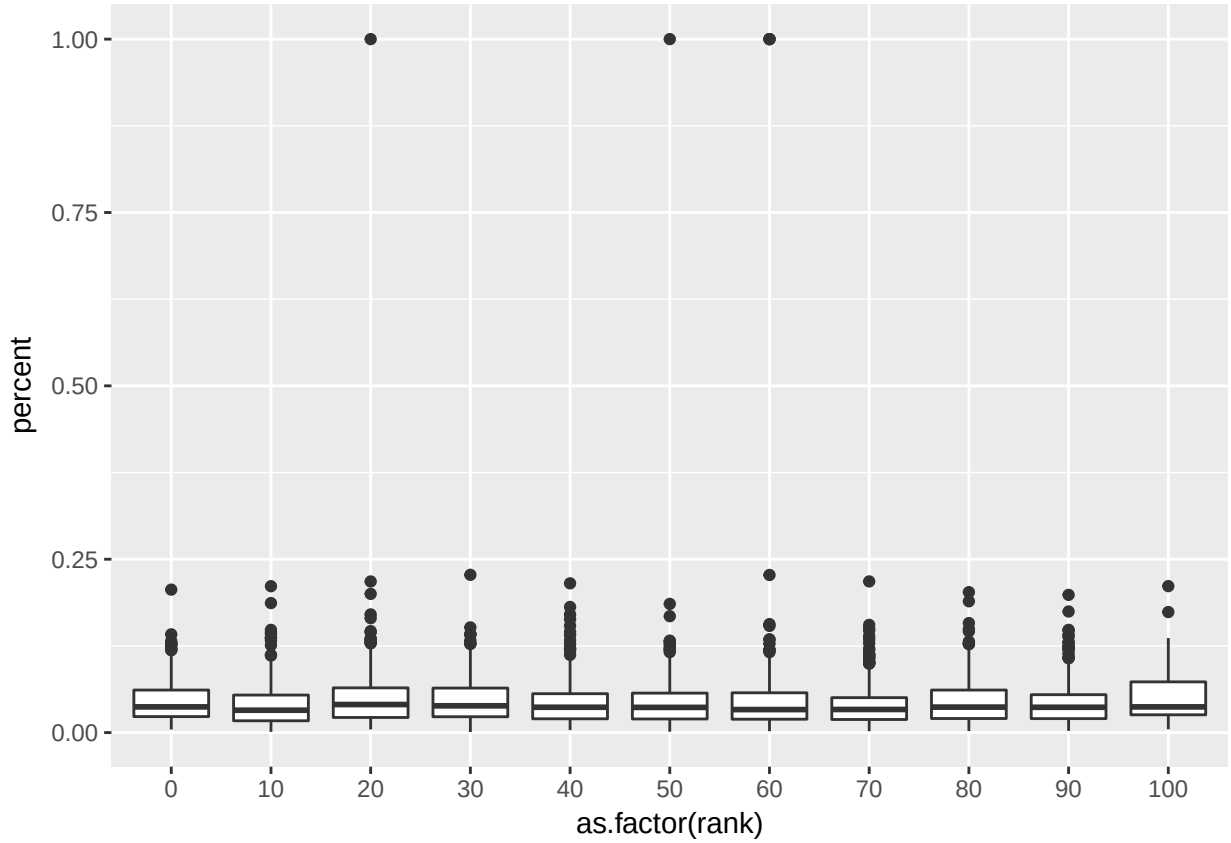
```
lyric_sentiment %>% count(Song, sentiment, total_words) %>%
                    ungroup() %>%
                    mutate(percent = n/total_words) %>%
                    filter(sentiment=="sadness") %>%
                    arrange(desc(percent))
```

```
## # A tibble: 2,904 x 5
##                 Song sentiment total_words     n   percent
##                <chr>     <chr>       <int> <int>     <dbl>
##  1         rack city   sadness         458   137 0.2991266
##  2        the stroke   sadness         279    52 0.1863799
```

```
##  3           bad boy   sadness         237    44 0.1856540
##  4         disturbia   sadness         956   164 0.1715481
##  5   ill tumble 4 ya   sadness         269    42 0.1561338
##  6      cruel summer   sadness         491    73 0.1486762
##  7        love shack   sadness         888   130 0.1463964
##  8             nasty   sadness         279    39 0.1397849
##  9     mad about you   sadness         155    20 0.1290323
## 10 youre only lonely   sadness         218    28 0.1284404
## # ... with 2,894 more rows
```

Let's check if the Billboard rank is related to sentiment

```
lyric_sentiment %>% filter(sentiment=="positive") %>%
                    count(Song, Rank, total_words) %>%
                    ungroup() %>%
                    mutate(percent = n/total_words,
                           rank = 10 * floor(Rank/10))%>%
                    ggplot(aes(as.factor(rank), percent)) +
                    geom_boxplot()
```

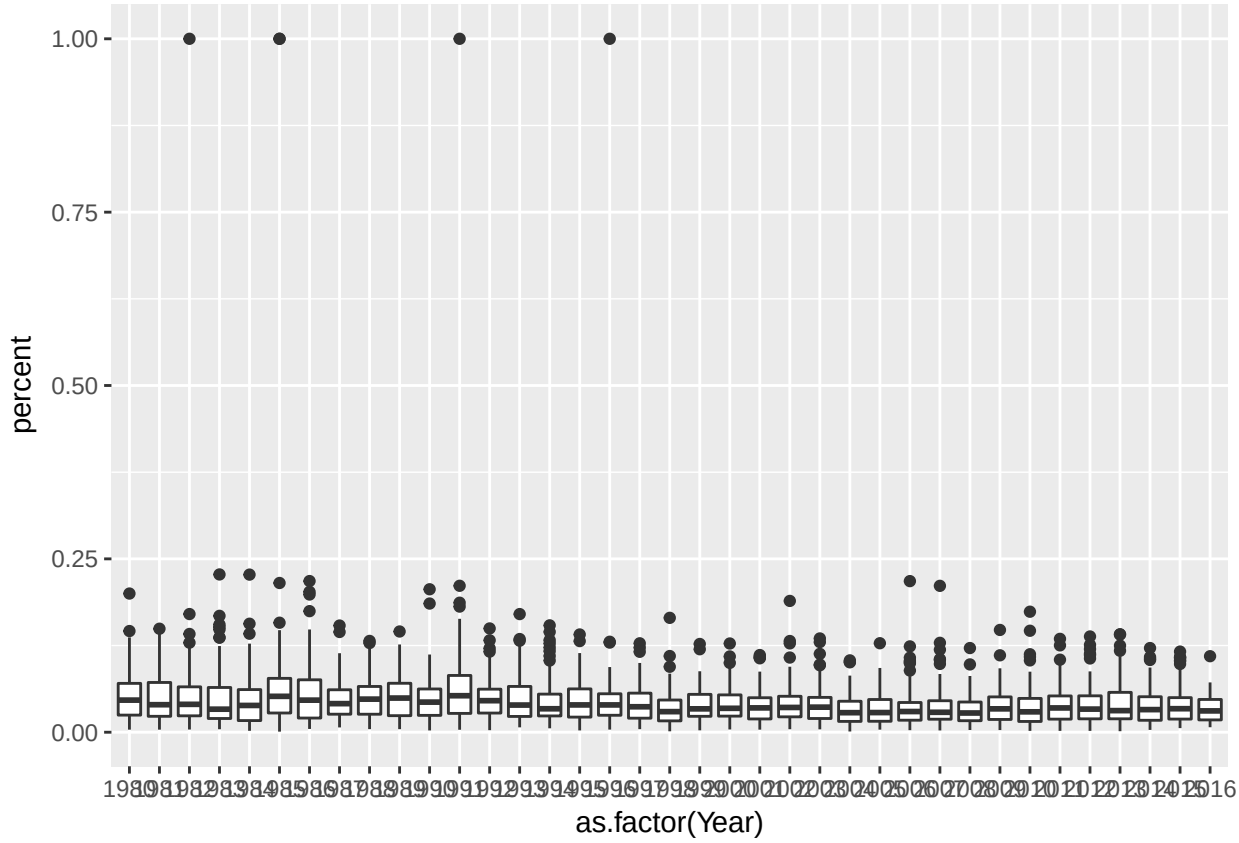



For positive sentiments the rank doesn't show any trend

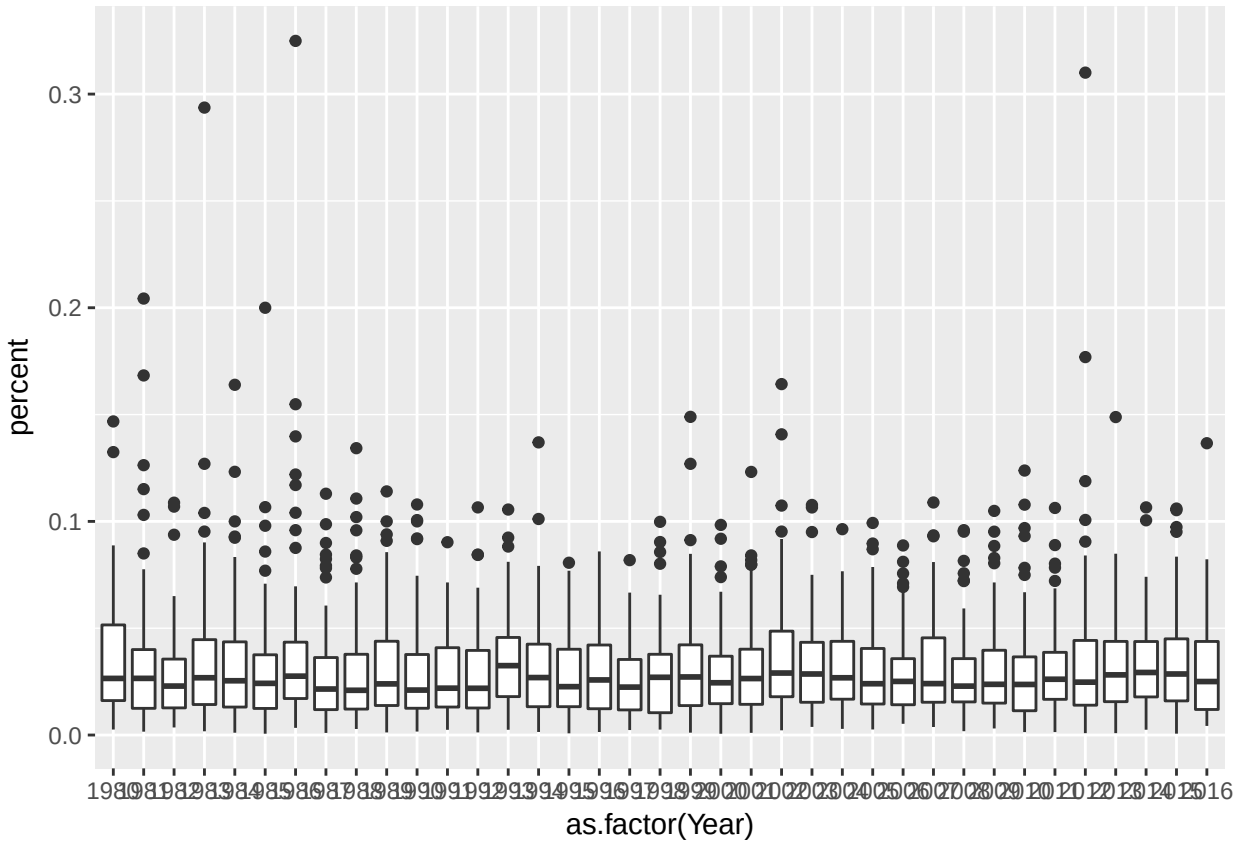
Are songs on the Billboard chart changing in their use of negative or positive words since 1980?

```
# For positive sentiments
 lyric_sentiment %>% filter(sentiment=="positive") %>%
```

```
                    count(Song, Year, total_words) %>%
                    ungroup() %>%
                    mutate(percent = n/total_words) %>%
                    ggplot(aes(as.factor(Year), percent)) +
                    geom_boxplot()
```




```
# For negative sentiments
 lyric_sentiment %>% filter(sentiment=="negative") %>%
                     count(Song, Year, total_words) %>%
                     ungroup() %>%
                     mutate(percent = n/total_words) %>%
                     ggplot(aes(as.factor(Year), percent)) +
                     geom_boxplot()
```




Let's try to model this sentiment

```
negative_data <- lyric_sentiment %>% filter(sentiment=="negative") %>%
                                     count(Song, Year, total_words) %>%
                                     ungroup() %>%
                                     mutate(percent = n/total_words)

negative_model <- lm(percent ~ Year, data=negative_data)
summary(negative_model)
```

```
##
## Call:
## lm(formula = percent ~ Year, data = negative_data)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.030448 -0.017032 -0.005708  0.010674  0.293843
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.146e-02  8.079e-02   0.637    0.524
## Year        -1.027e-05  4.044e-05  -0.254    0.799
##
## Residual standard error: 0.0247 on 3401 degrees of freedom
## Multiple R-squared:  1.898e-05,  Adjusted R-squared:  -0.000275
## F-statistic: 0.06455 on 1 and 3401 DF,  p-value: 0.7995
```

The p-value is $0.79 > 0.05$ so we can say Year for negative sentiment doesn't play a important role

Let's do the same for positive

```
positive_data <- lyric_sentiment %>% filter(sentiment=="positive") %>%
                                      count(Song, Year, total_words) %>%
                                      ungroup() %>%
                                      mutate(percent = n/total_words)

positive_model <- lm(percent ~ Year, data=positive_data)
summary(positive_model)
```

```
##
## Call:
## lm(formula = percent ~ Year, data = positive_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.05256 -0.02351 -0.00766  0.01255  0.95382
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.299e+00  1.544e-01   8.414  < 2e-16 ***
## Year        -6.276e-04  7.727e-05  -8.122 6.29e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04784 on 3466 degrees of freedom
## Multiple R-squared:  0.01868,    Adjusted R-squared:  0.0184
## F-statistic: 65.97 on 1 and 3466 DF,  p-value: 6.293e-16
```

In this case p-value is significant, and Year is a important variable

Let's add sentiment as variable

```
data_mod <- lyric_sentiment %>% count(Song, Year, sentiment, total_words) %>%
                                      ungroup() %>%
                                      mutate(percent = n/total_words)

data_model <- lm(percent ~ Year + sentiment, data=data_mod)
summary(data_model)
```

```
##
## Call:
## lm(formula = percent ~ Year + sentiment, data = data_mod)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.04669 -0.01246 -0.00522  0.00660  0.95460
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)            4.063e-01  2.603e-02  15.605  < 2e-16 ***
## Year                  -1.945e-04  1.303e-05 -14.932  < 2e-16 ***
```

```
## sentimentanticipation  5.206e-03  6.089e-04   8.551  < 2e-16 ***
## sentimentdisgust      -3.472e-03  6.362e-04  -5.458 4.84e-08 ***
## sentimentfear          5.209e-04  6.168e-04   0.845   0.3984
## sentimentjoy           1.446e-02  6.086e-04  23.752  < 2e-16 ***
## sentimentnegative      1.330e-02  6.076e-04  21.886  < 2e-16 ***
## sentimentpositive      2.739e-02  6.049e-04  45.282  < 2e-16 ***
## sentimentsadness       1.747e-03  6.149e-04   2.841   0.0045 **
## sentimentsurprise     -4.117e-03  6.255e-04  -6.581 4.73e-11 ***
## sentimenttrust         3.395e-03  6.115e-04   5.552 2.84e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02455 on 32370 degrees of freedom
## Multiple R-squared:  0.134,  Adjusted R-squared:  0.1337
## F-statistic: 500.7 on 10 and 32370 DF,  p-value: < 2.2e-16
```

Fear is not significant variable for the Billboard rank