Lab 1

Lucy Whitmore

2023-01-23

Data

We'll work with the #tidytuesday data for 2019, specifically the #rstats dataset, containing nearly 500,000 tweets over a little more than a decade using that hashtag.

The data is in under Dataset tab of Week 3 module on Canvas.

You can import the dataset using the code below.

```r
d <- rio::import(here::here("data", "rstats_tweets.rds"),
                 setclass = "tbl_df")
```

If you need help with processing text data, please revisit the notebook introduced in Week 1.

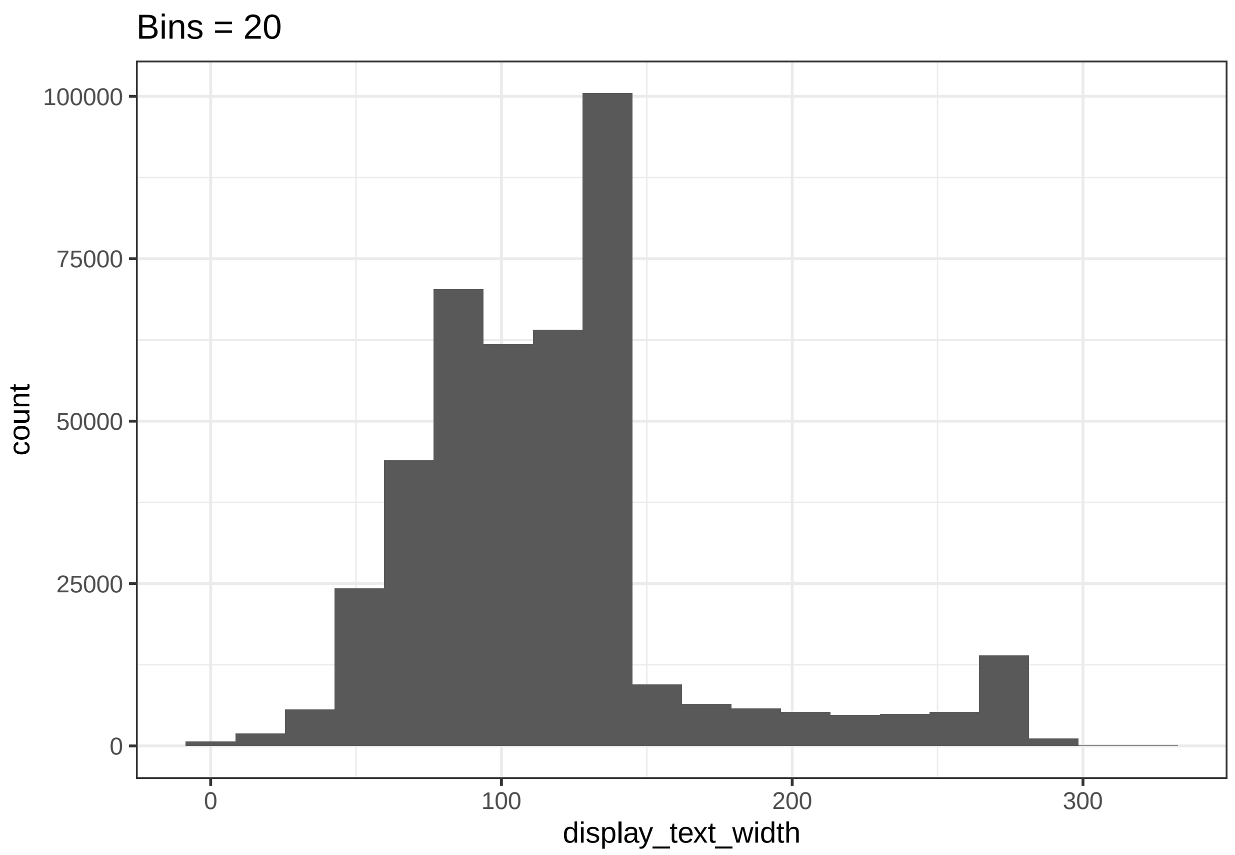
https://www.kaggle.com/code/uocoeeds/introduction-to-textual-data

Histogram and Density plots

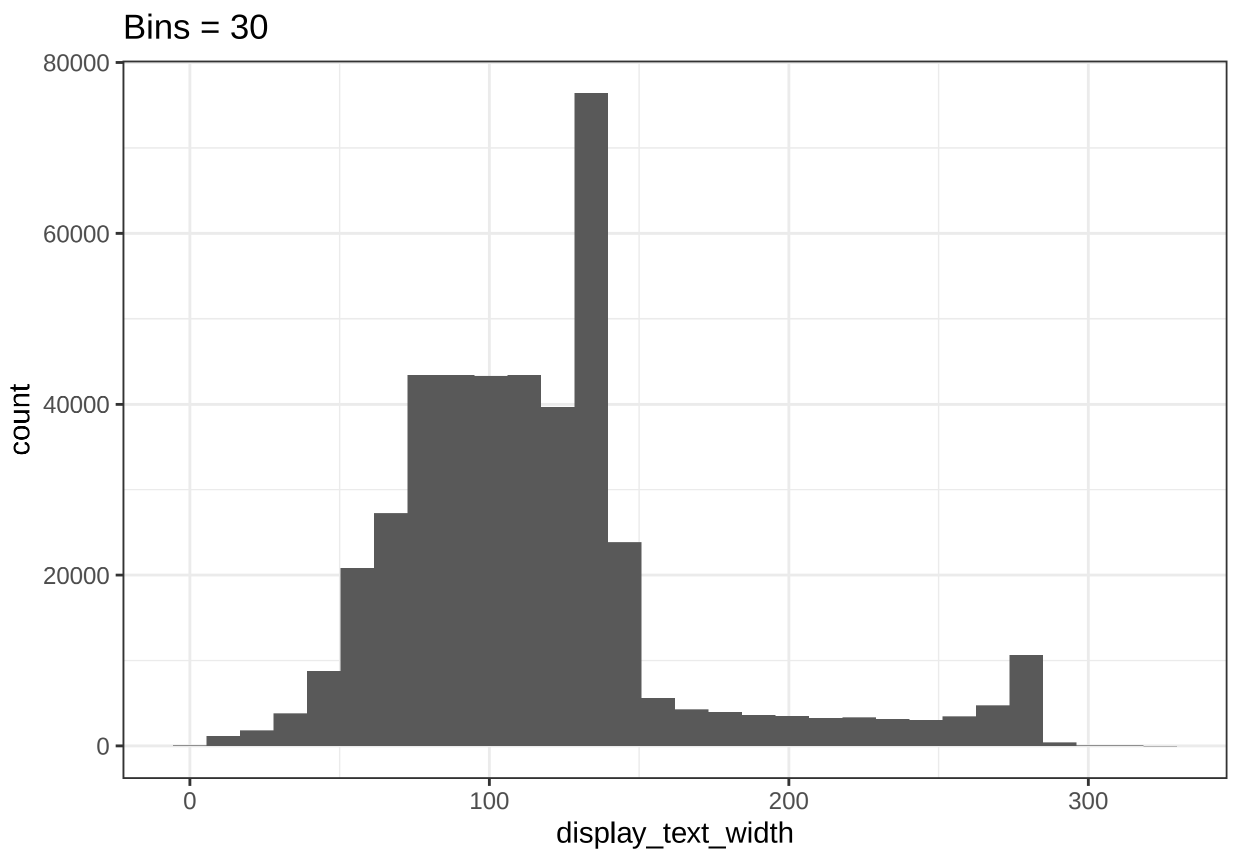
1. Create a histogram the column `display_text_width` using the `ggplot2` package and `geom_histogram()` function. Try at least four different numbers of bins (e.g., 20, 30, 40, 50) by manipulating the `bins=` argument. Select what you think best represents the data for each. Provide a brief justification for your decision. For all plots you created, change the default background color from grayish to white.

In this case, I think 50 bins is the best representation of the data. At 20 bins and 30, it's hard to see the patterns in the data, as everything is being lumped in together. At 60 bins, there's one bin in particular that's really changing the y-axis limits, which makes it difficult to see and interpret the other values, as all the other bins become squished. 40 and 50 bins are both good for seeing the overall distribution without squishing the y-axis. However, I prefer 50 bins, as it's helpful for visualing the specific bins that are driving the peaks in the distribution.

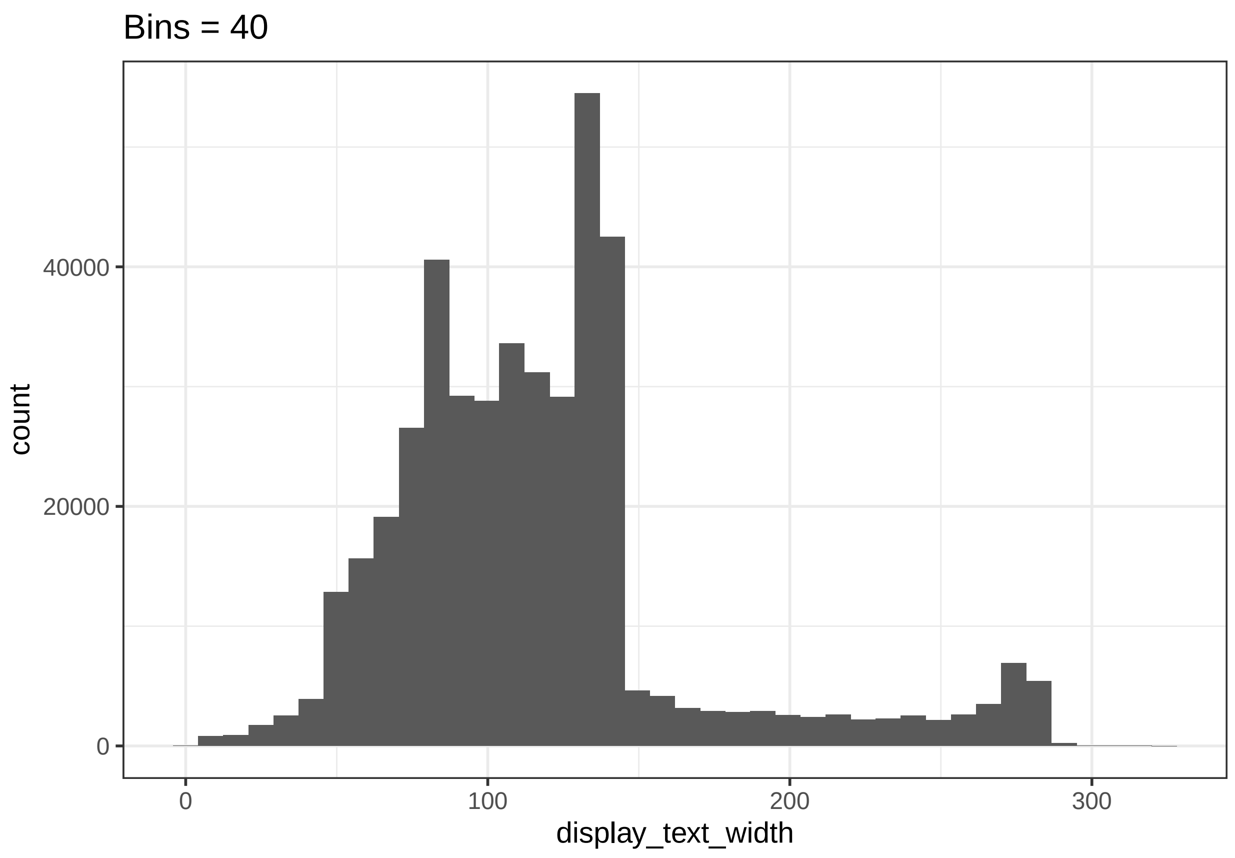
```r
ggplot(d, aes(x=display_text_width)) +
  geom_histogram(bins=20) +
  labs(title = "Bins = 20") +
  theme_bw()
```



```
ggplot(d, aes(x=display_text_width)) +
  geom_histogram(bins=30) +
  labs(title = "Bins = 30") +
  theme_bw()
```

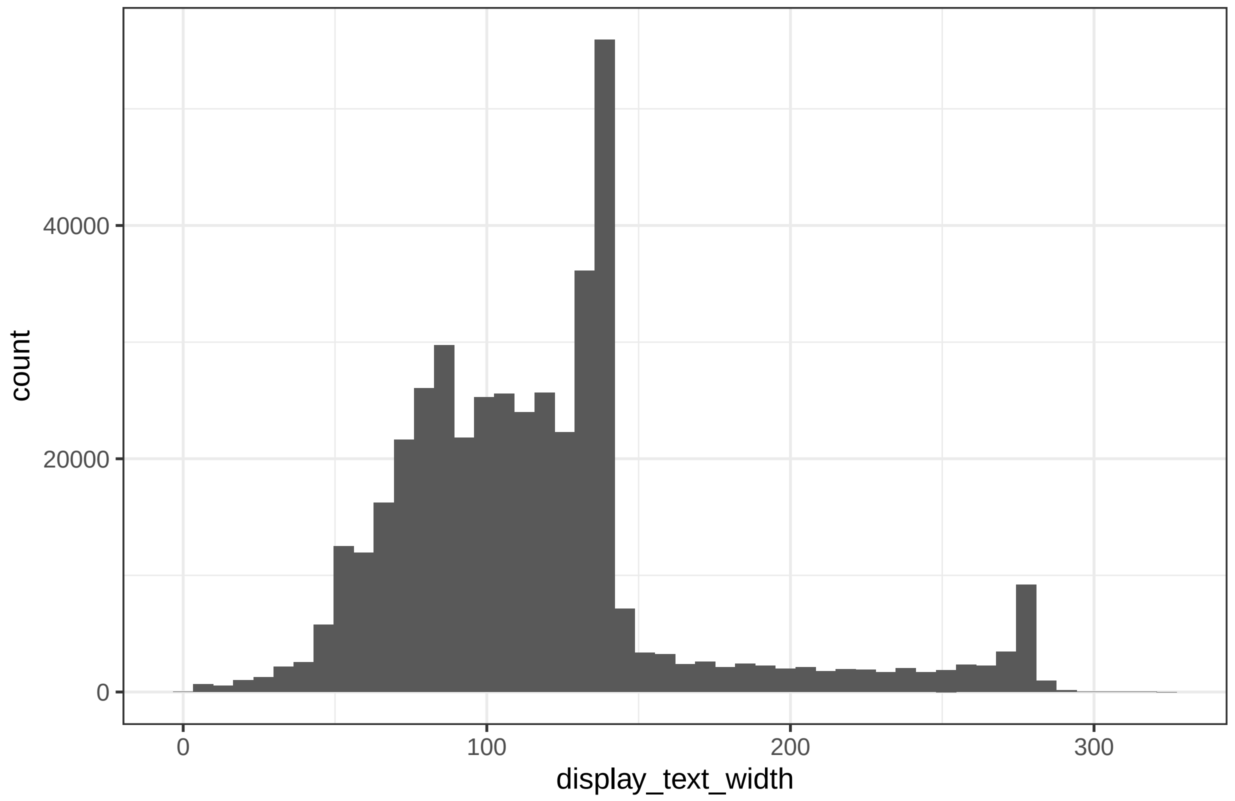


```
ggplot(d, aes(x=display_text_width)) +
  geom_histogram(bins=40) +
  labs(title = "Bins = 40") +
  theme_bw()
```

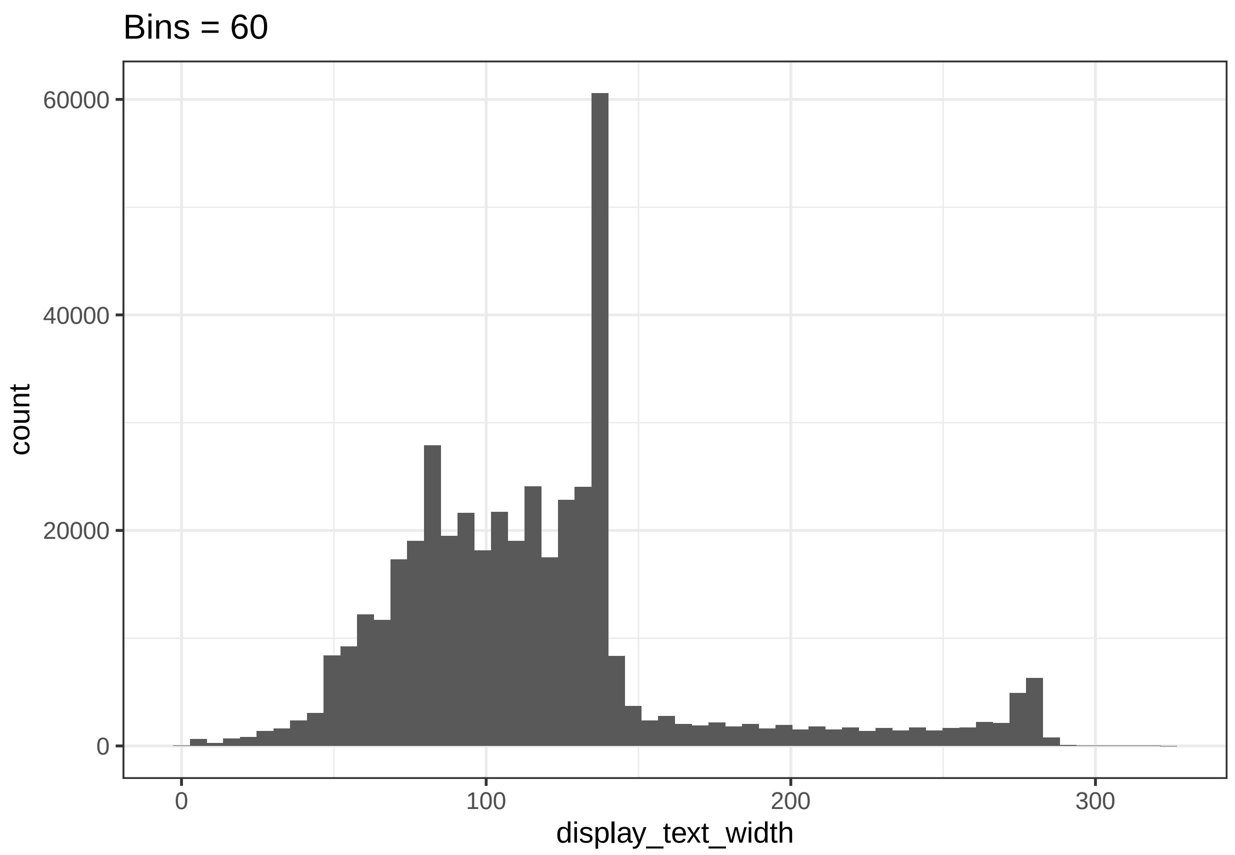


```
ggplot(d, aes(x=display_text_width)) +
  geom_histogram(bins=50) +
  labs(title = "Bins = 50") +
  theme_bw()
```

Bins = 50



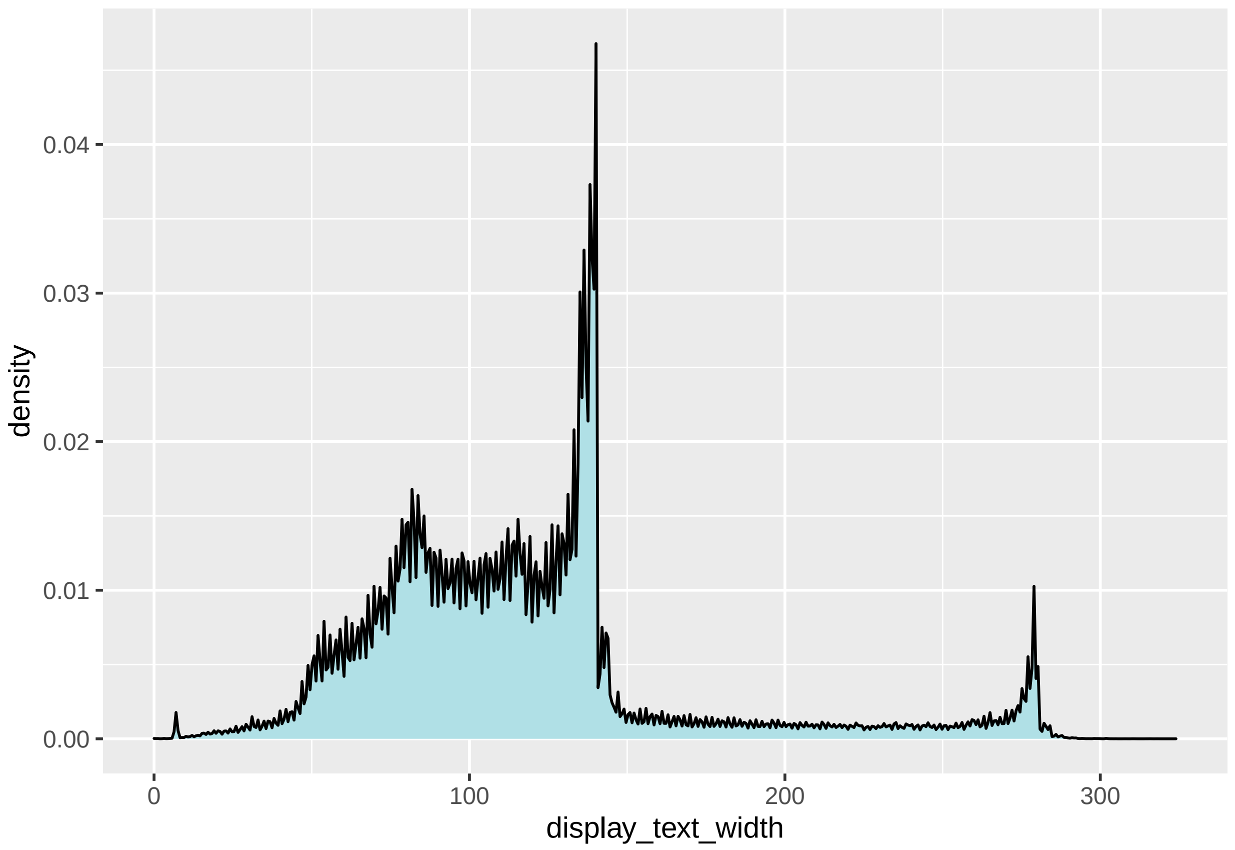
```
ggplot(d, aes(x=display_text_width)) +
 geom_histogram(bins=60) +
labs(title = "Bins = 60") +
 theme_bw()
```



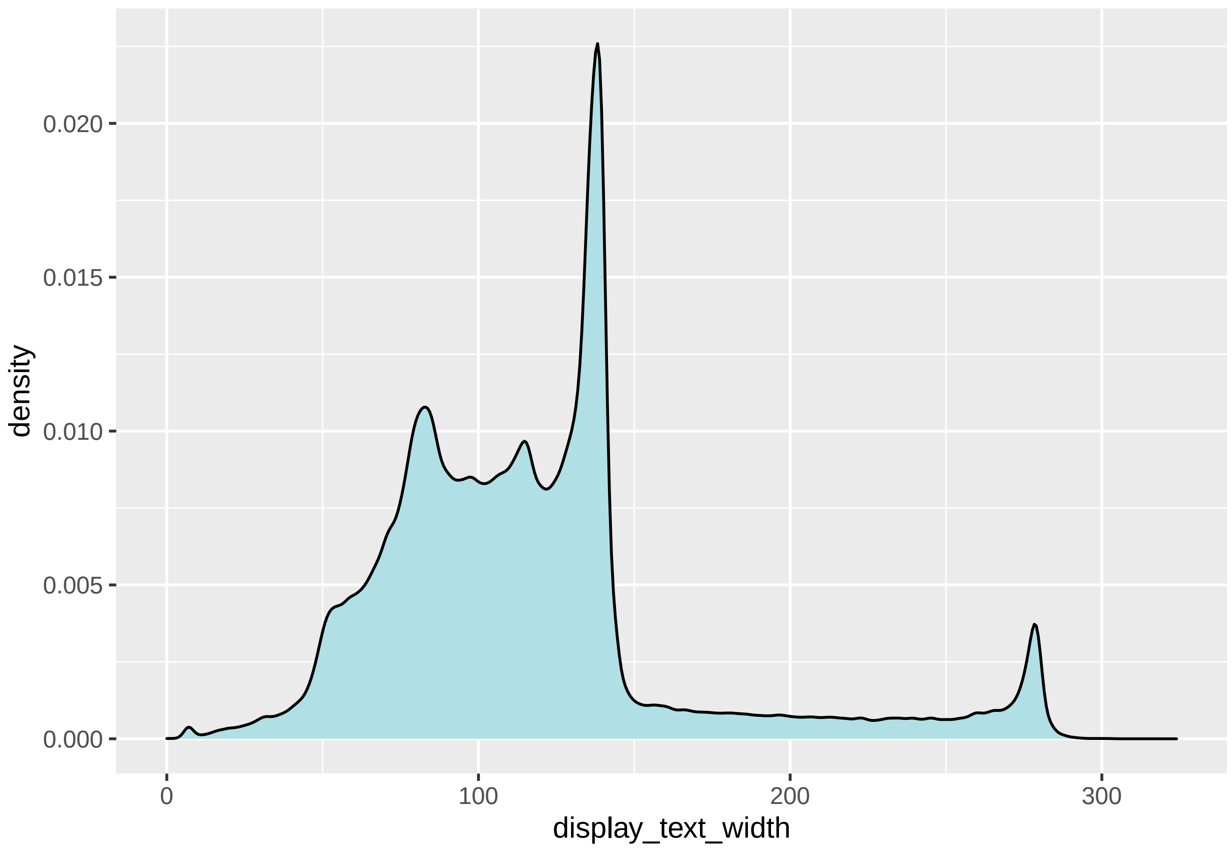
2. Create a density plot for the column `display_text_width` using the `ggplot2` package and `geom_density()` function. Fill the inside of density plot with a color using the `fill=` argument. Try at least four different numbers of smoothing badwith (e.g., 0.2, 1.5, 3, 5) by manipulating the `bw=` argument. Select what you think best represents the data for each. Provide a brief justification for your decision.

In this case, I think a smoothing bandwidth of 3 is best. At lower bandwidths, especially .2, it's difficult to see the distribution, as there is so much variation/jitter in the line. However, at higher bandwidths, like 10, the distribution of the data is smoothed too much, which obscures the true pattern and variation in the data.

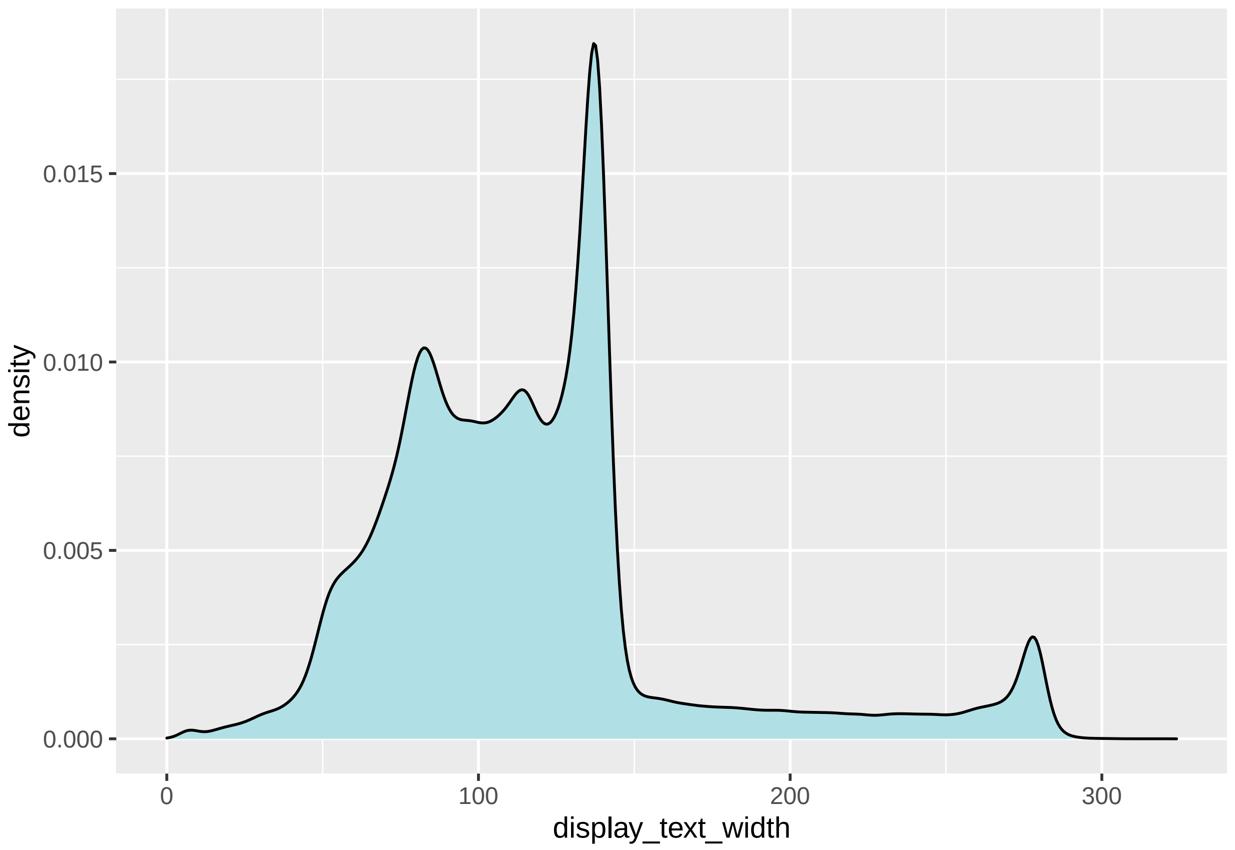
```
ggplot(d, aes(x=display_text_width)) +
  geom_density(bw=0.2, fill="#b0e0e6")
```



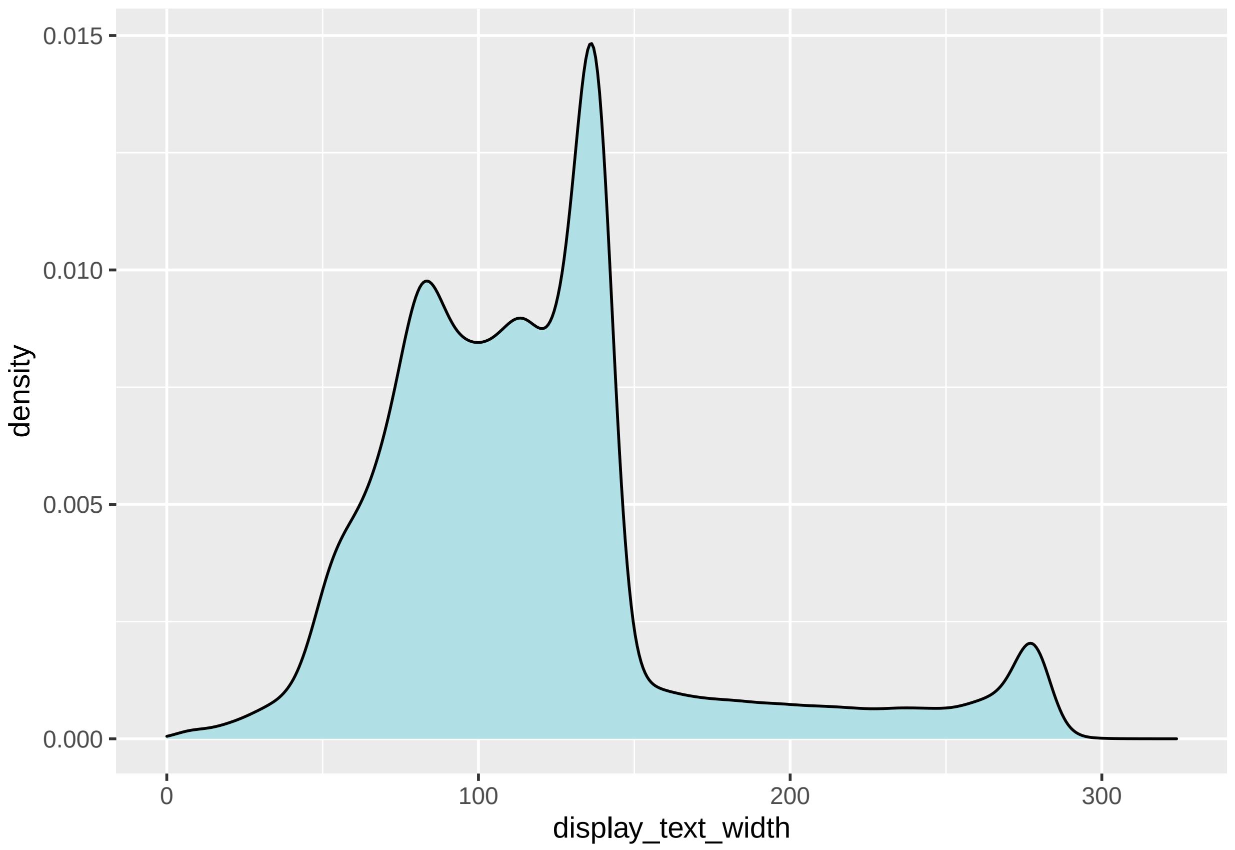
```
ggplot(d, aes(x=display_text_width)) +
  geom_density(bw=1.5, fill = "#b0e0e6")
```



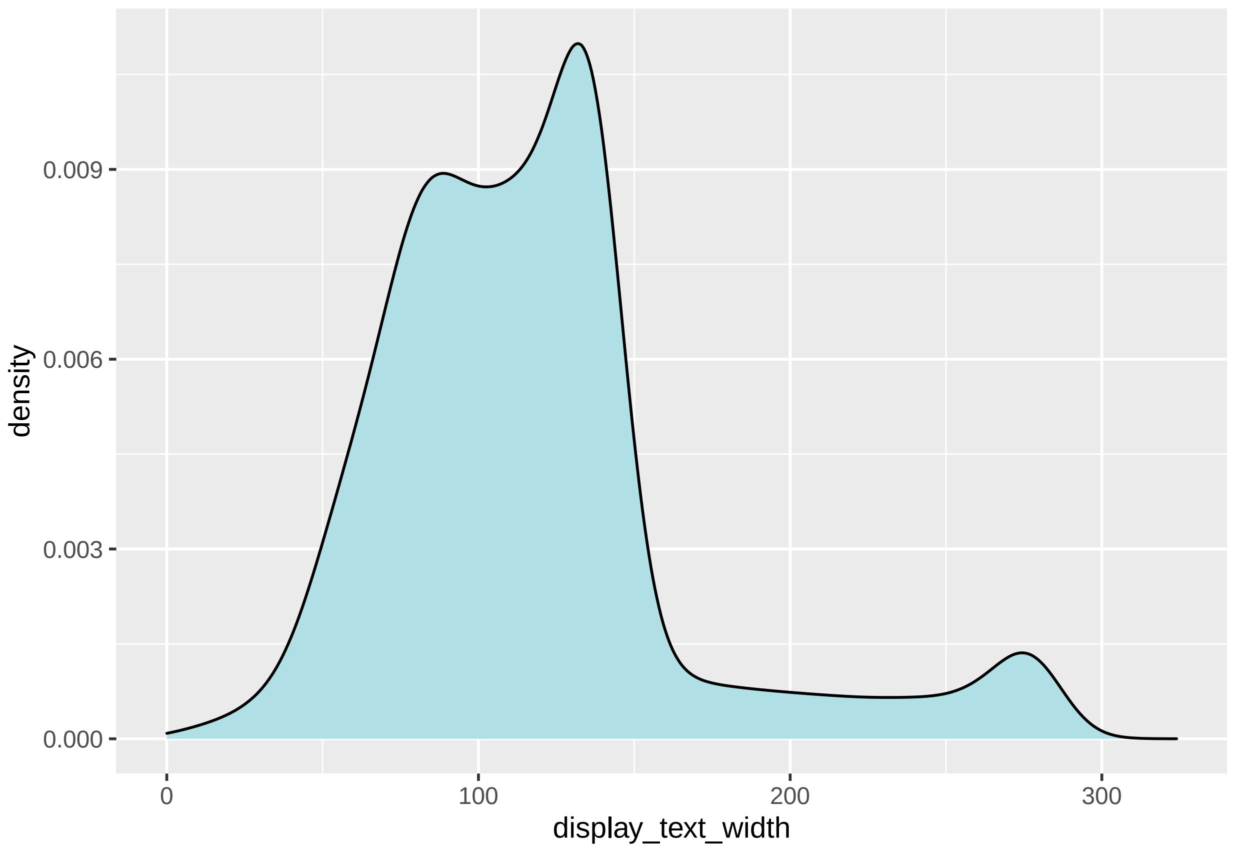
```
ggplot(d, aes(x=display_text_width)) +
  geom_density(bw=3, fill = "#b0e0e6")
```



```
ggplot(d, aes(x=display_text_width)) +
  geom_density(bw=5, fill = "#b0e0e6")
```



```
ggplot(d, aes(x=display_text_width)) +
  geom_density(bw=10, fill = "#b0e0e6")
```



Barplot

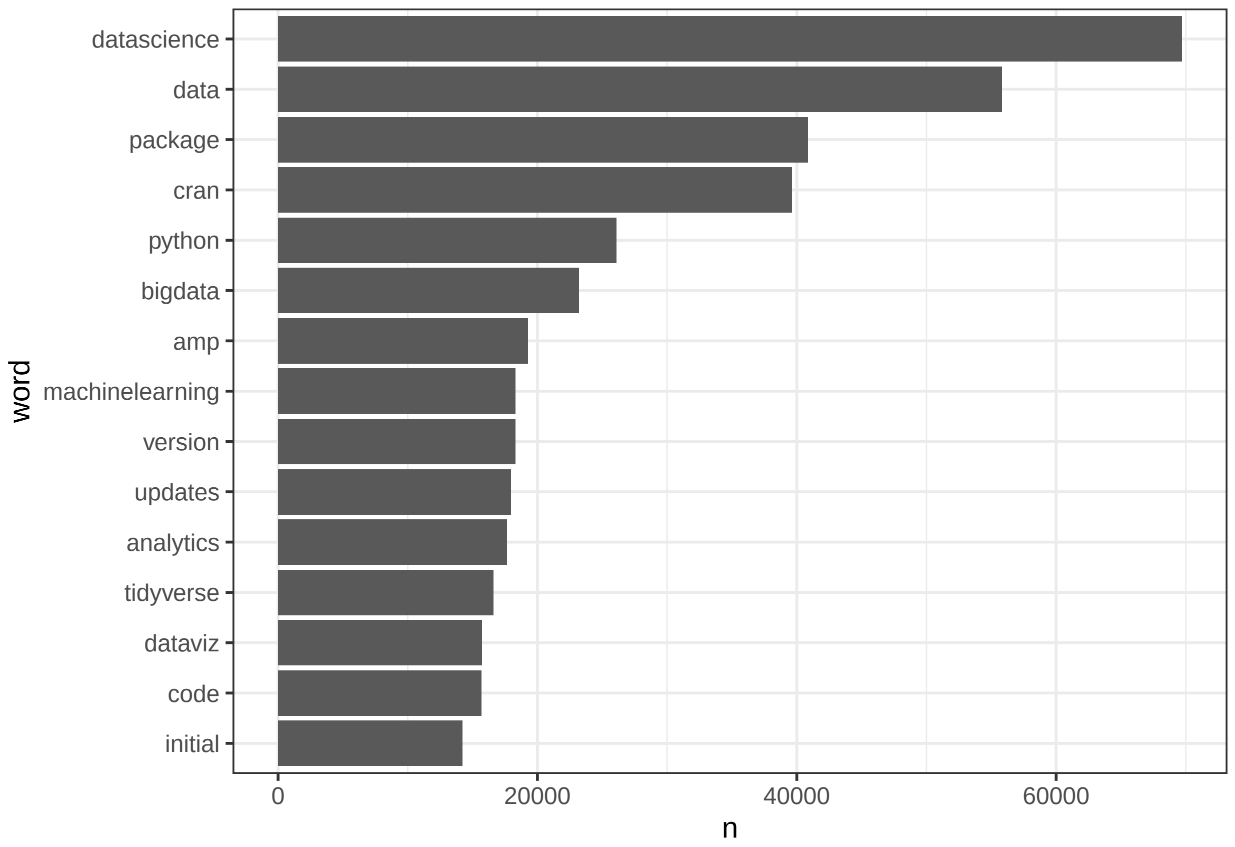
3. Using the information `text` column, create the following figure of the 15 most common words represented in these posts by using the `ggplot2()` package and `geom_col()` function. Remove the stop words, and also exclude the words such as 't.co','https','http','rt','rstats'.

```
rstats_tidy_words <- d %>%
  unnest_tokens(word, text) %>%
  select(word)

rstats_tidy_words_cleaned <- rstats_tidy_words %>%
  anti_join(stop_words) %>%
  filter(!(word %in% c('t.co','https','http','rt','rstats'))) %>%
  count(word, sort = TRUE) %>%
  mutate(word = reorder(word, n)) %>% # make y-axis ordered by n
  slice(1:15) # select only the first 15 rows
```

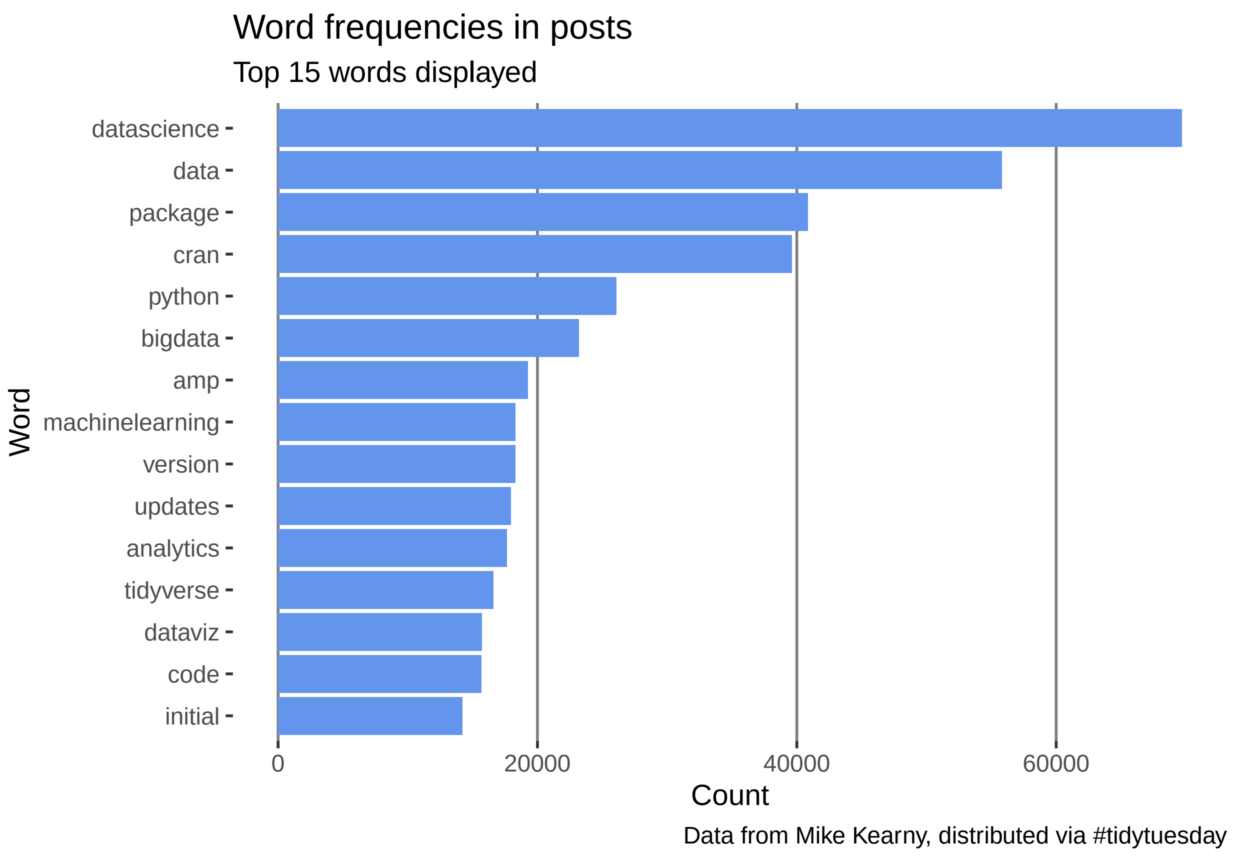
```
## Joining, by = "word"
```

```
rstats_tidy_words_cleaned %>%
ggplot(aes(n, word)) +
    geom_col() +
  theme_bw()
```



4. Style the plot so it (mostly) matches the below. It does not need to be exact, but it should be close.

```
rstats_tidy_words_cleaned %>%
ggplot(aes(n, word)) +
  geom_col(fill="cornflowerblue") +
  labs(title = "Word frequencies in posts", subtitle = "Top 15 words displayed", caption = "Data from M
  theme(panel.background = element_rect(fill = NA),
        panel.grid.major = element_line(colour = "grey50"),
        panel.grid.major.y = element_blank(),
  panel.grid.minor.y = element_blank())
```


Word frequencies in posts
Top 15 words displayed
datascience
data
package
cran
python
bigdata
amp
machinelearning
version
updates
analytics
tidyverse
dataviz
code
initial
Word
0
20000
40000
60000
Count
Data from Mike Kearny, distributed via #tidytuesday