Lab 3 complete

Lawrence Y. Tello

11/11/2020

Welcome to Lab 3 !!!

Great article from (Harvard Business Review about data visualization that really work) [https://hbr.org/2016/06/visualizations-that-really-work].

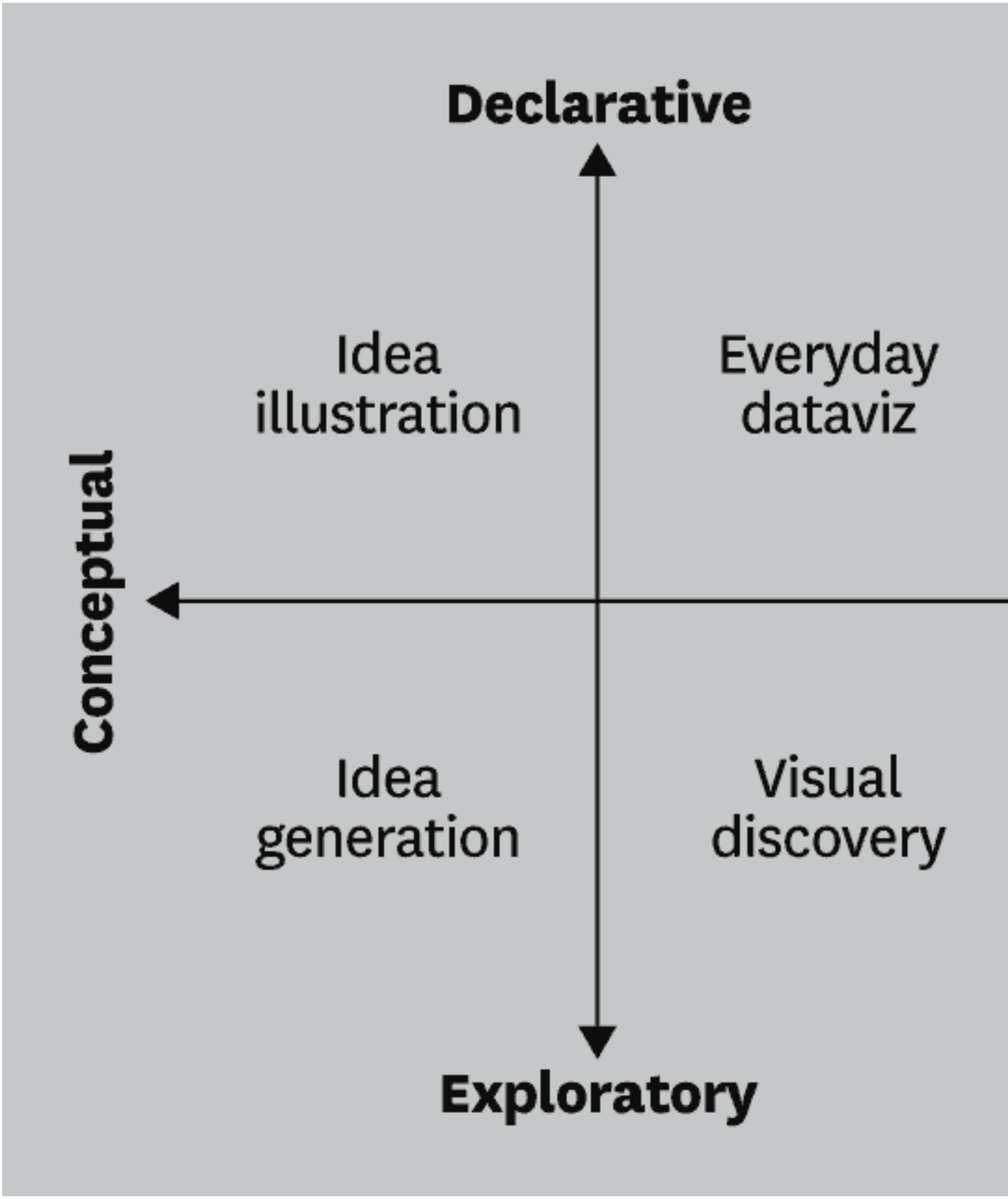
The (R Graph Gallery)[https://www.r-graph-gallery.com/index.html] is an excellent resource for data visualization ideas AND code to accomplish these visualizations.

(A nice website detailing many of the tools we've learned already for creating nice tables)[https://rfortherestofus.com/2019/11/how-to-make-beautiful-tables-in-r/].

(Our World in Data has some great visualizations)[https://ourworldindata.org/].

If you're working on your Desktop and want to change up your RStudio Theme, (check out these themes made by the community)[https://tmtheme-editor.herokuapp.com/#!/editor/theme/Monokai].

```
knitr::include_graphics('data/hbr_graph.png')
```


Declarative
Idea illustration
Everyday dataviz
Conceptual
Idea generation
Visual discovery
Exploratory

Today's goals:

Part 1:

- rotating graphs
- modifying axis limits
- ordering by value
- creating text labels
- adding annotations
- saving plots

Part 2:

- scaling axis (values/time)
- hiding x and y elements

Part 3:

- using functions to "automate" graph creation
- detailed modifying of graph element

Part 1

- rotating graphs
- modifying axis limits
- ordering by value
- creating text labels
- adding annotations

https://yutannihilation.github.io/allYourFigureAreBelongToUs/ggthemes/

```
gen_plot <- read_csv("data/sample_plot.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   X1 = col_double(),
##   patient_id = col_double(),
##   age = col_double(),
##   race_ethnicity = col_character(),
##   gender_identity = col_character(),
##   height = col_double(),
##   weight = col_double(),
##   visit_id = col_double(),
##   date = col_character()
## )
```

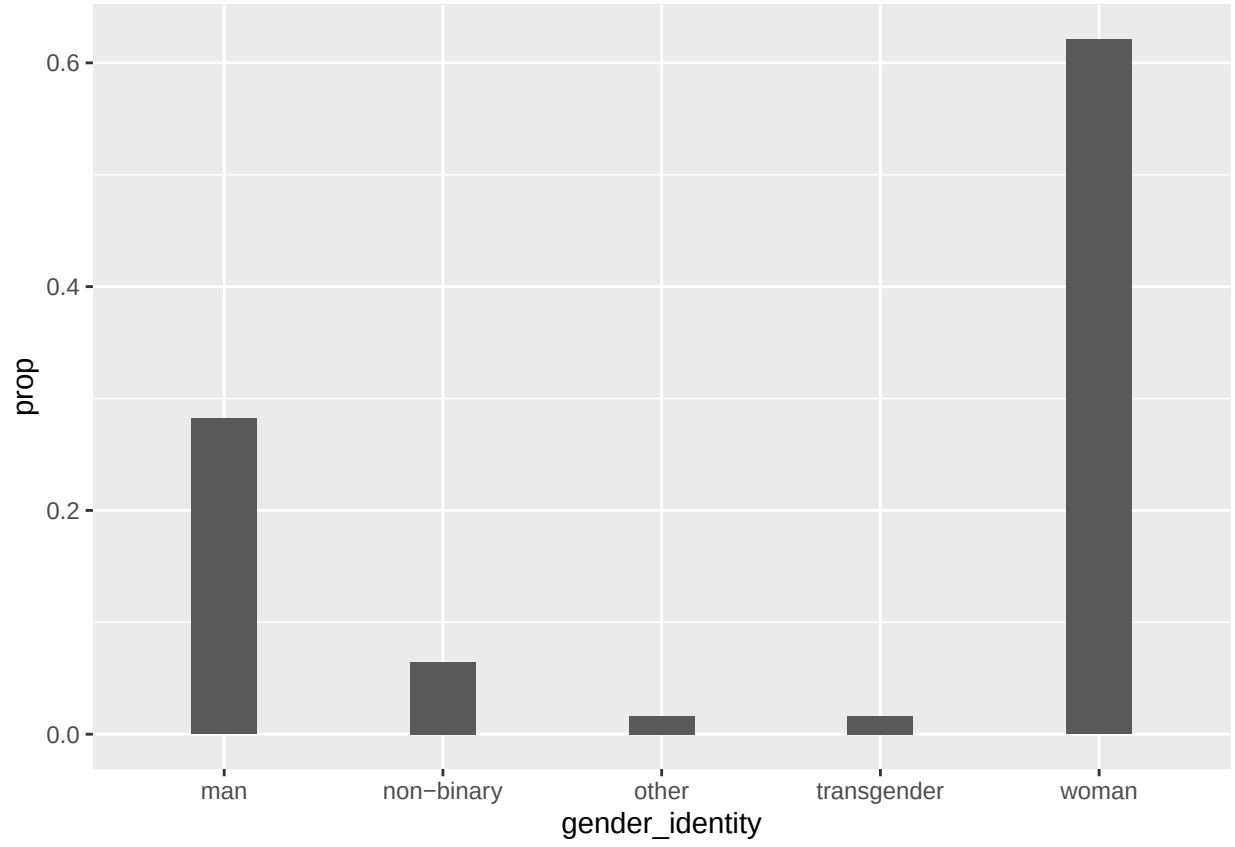
```
plot <- gen_plot %>%
  mutate( count = 1 ) %>%
  select( gender_identity, count ) %>%
  drop_na() %>%
  group_by(gender_identity) %>%
  summarize( num = n() )
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

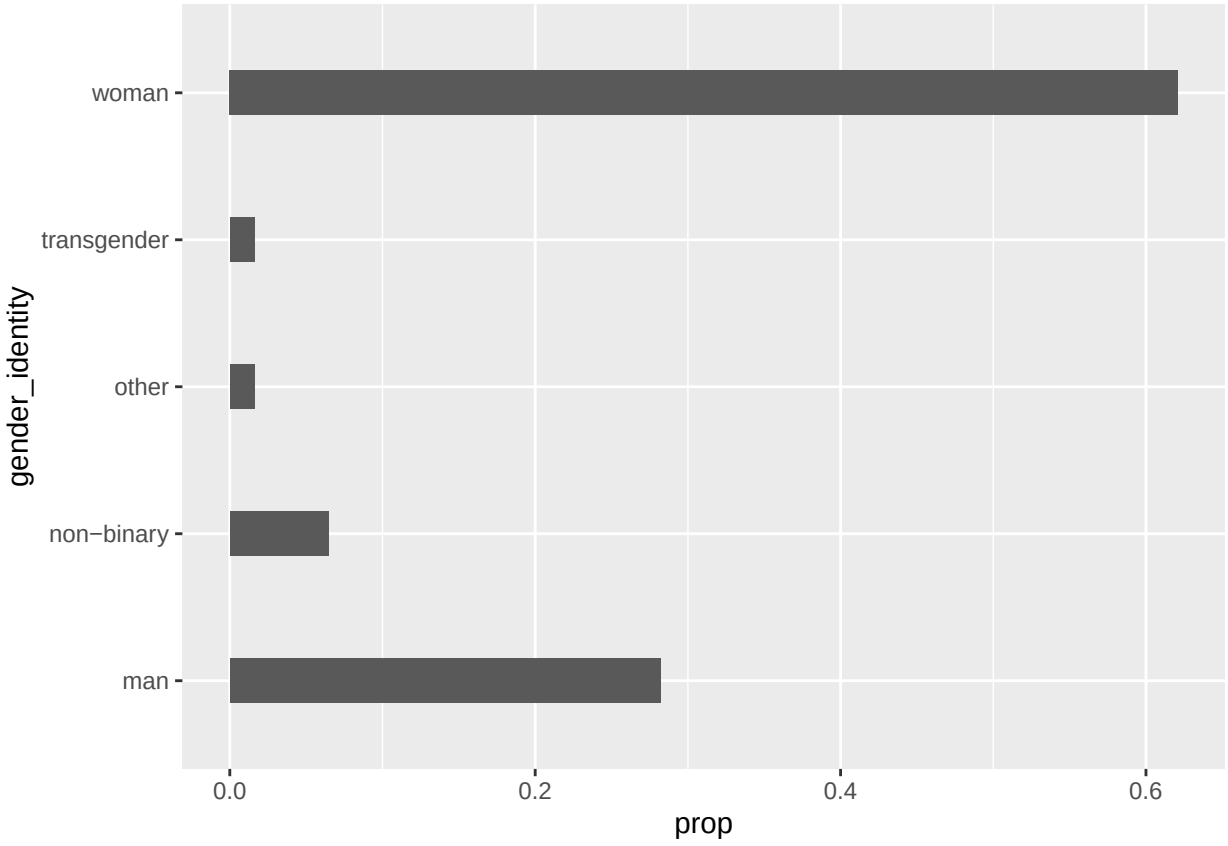
```
plot$total <- sum( plot$num )
plot$prop <- plot$num / plot$total
sum ( plot$prop )
```

```
## [1] 1
```

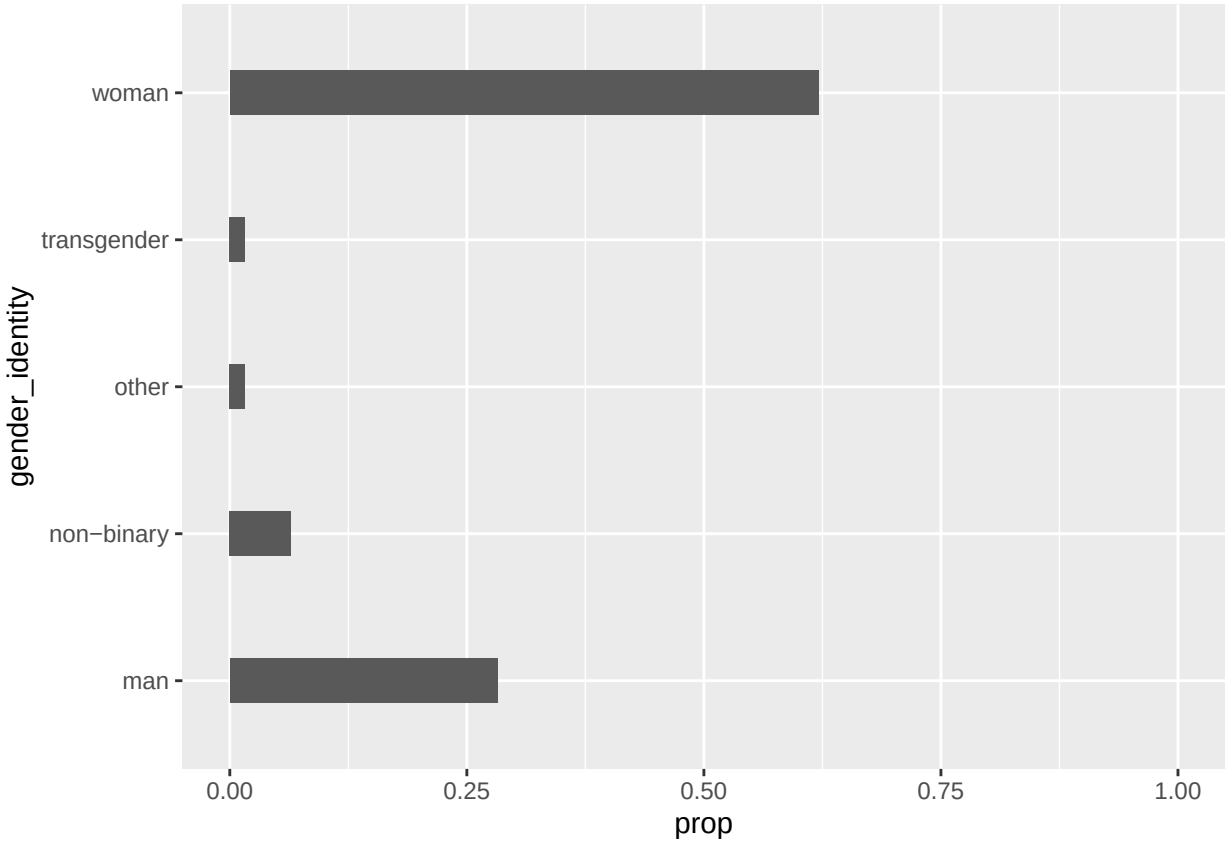
```
ggplot(plot, aes( x = gender_identity, y = prop ) ) +
  geom_bar( stat = "identity", width = 0.3 )
```



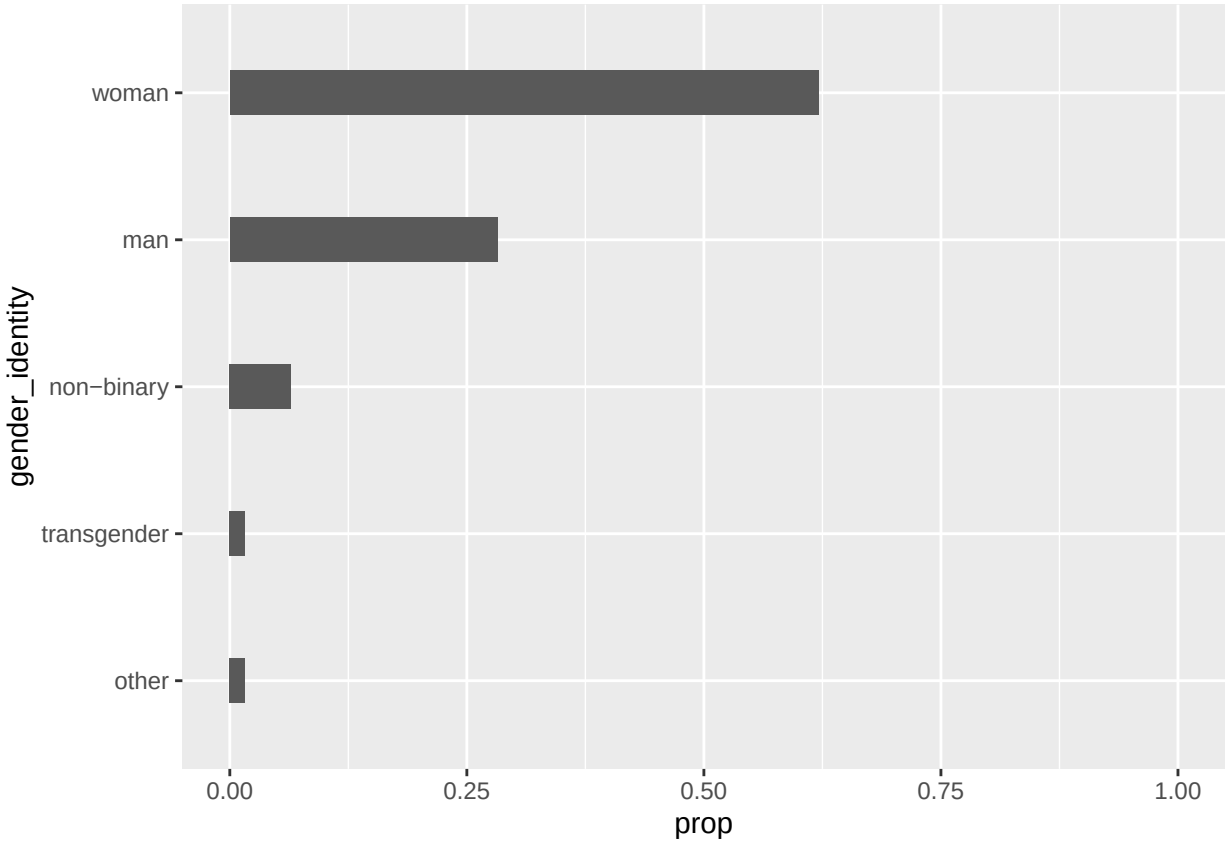

```
ggplot(plot, aes( x = gender_identity, y = prop ) ) +
  geom_bar( stat = "identity", width = 0.3 ) +
  coord_flip()
```



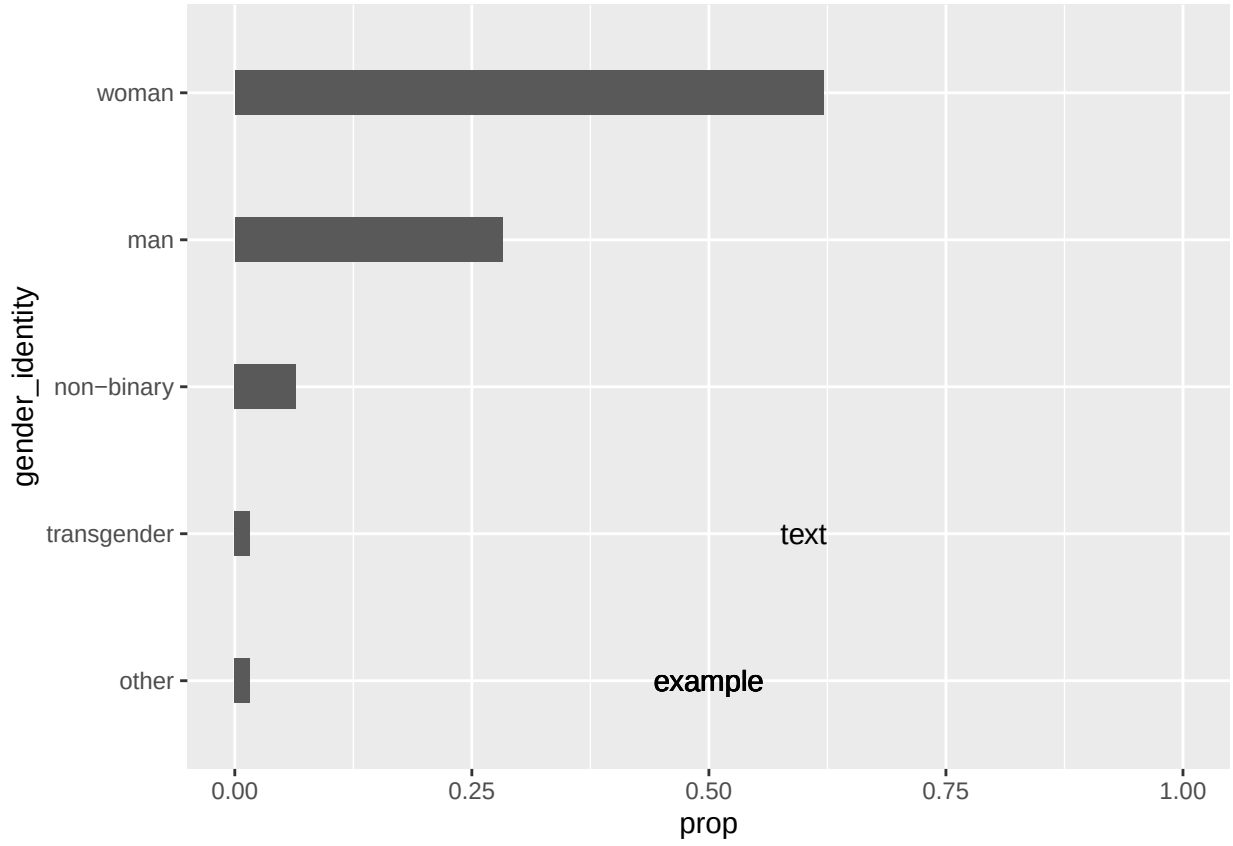

```
ggplot(plot, aes( x = gender_identity, y = prop ) ) +
  geom_bar( stat = "identity", width = 0.3 ) +
  coord_flip() +
  ylim( 0 , 1 )
```



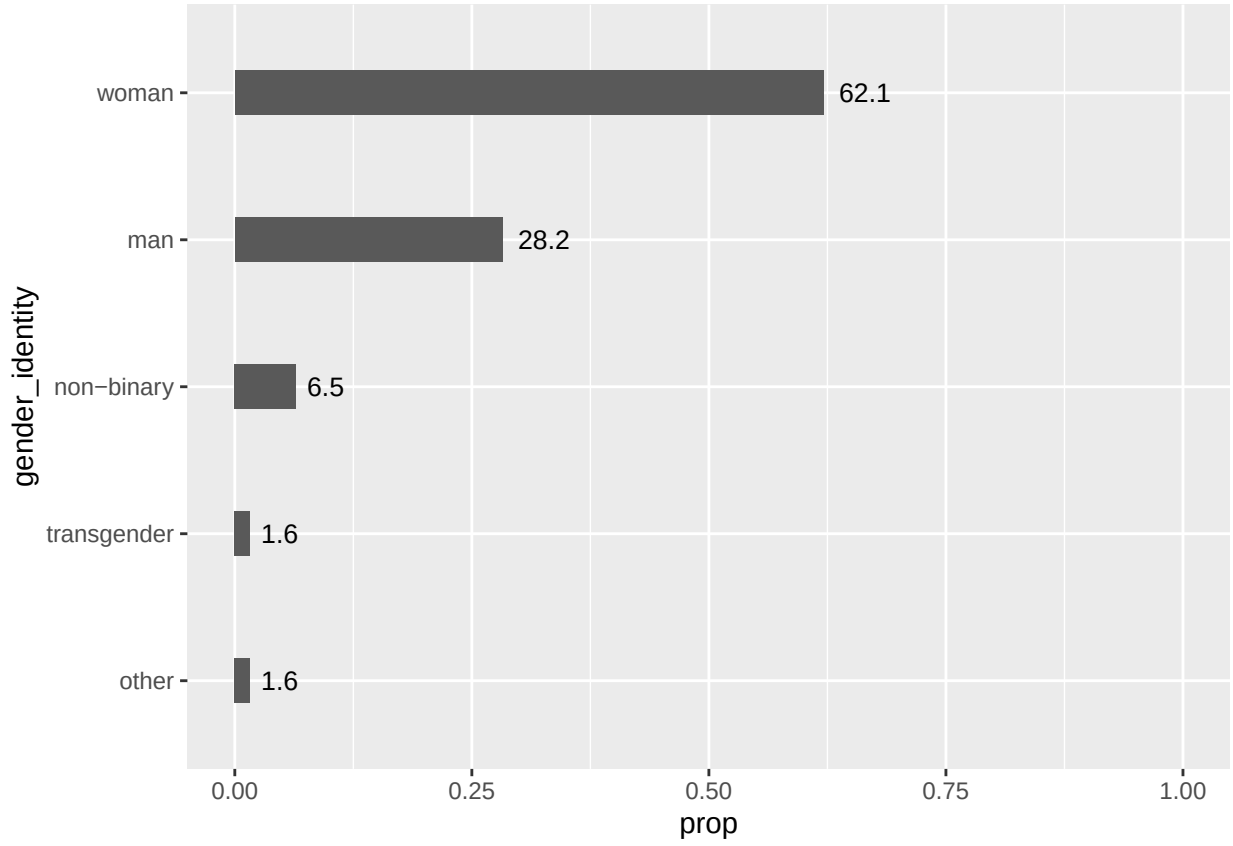

```
plot <- plot %>%
  mutate( gender_identity = factor ( gender_identity,
                                     levels = gender_identity[ order( num ) ] ))

ggplot(plot, aes( x = gender_identity, y = prop ) ) +
  geom_bar( stat = "identity", width = 0.3 ) +
  coord_flip() +
  ylim( 0 , 1 )
```



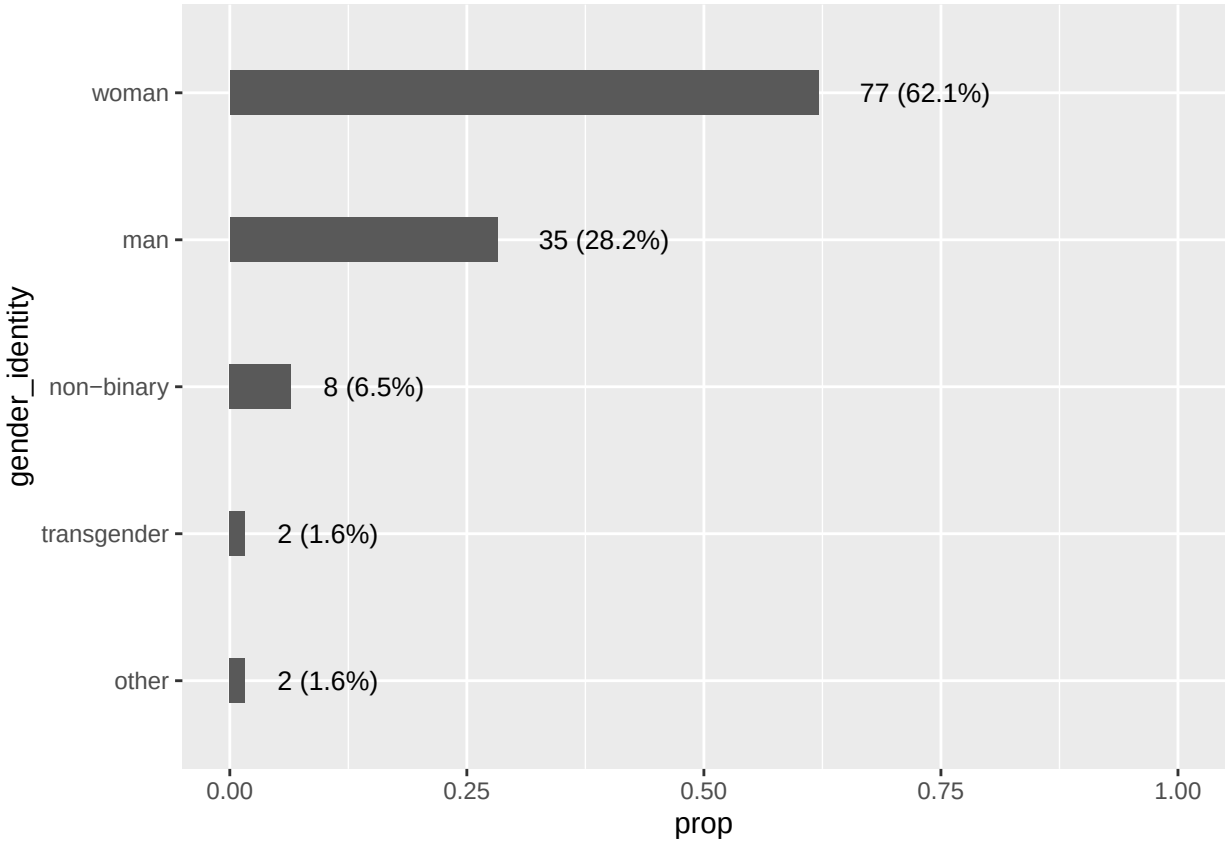

```
ggplot(plot, aes( x = gender_identity, y = prop ) ) +
  geom_bar( stat = "identity", width = 0.3 ) +
  coord_flip() +
  ylim( 0 , 1 ) +
  geom_text( aes( x = 1, y = .5, label = "example")) +
  annotate("text", x = 2, y = .6, label = "text")
```



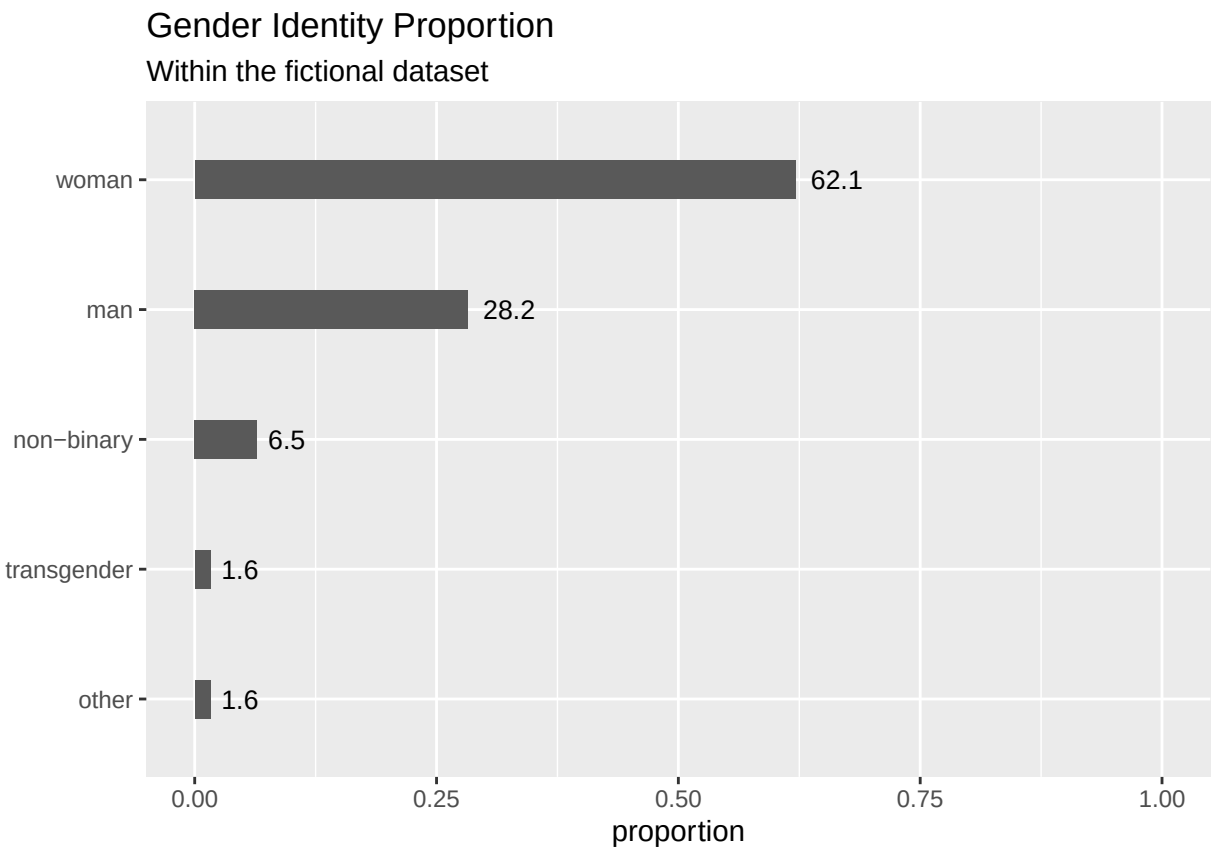

```
plot$perc <- as.character( round ( plot$prop, 3 ) * 100 )

ggplot(plot, aes( x = gender_identity, y = prop ) ) +
  geom_bar( stat = "identity", width = 0.3 ) +
  coord_flip() +
  ylim( 0 , 1 ) +
  geom_text( aes( x = gender_identity, y = prop, label = perc ),
             size = 3.5, vjust = .5, hjust = -.3 )
```



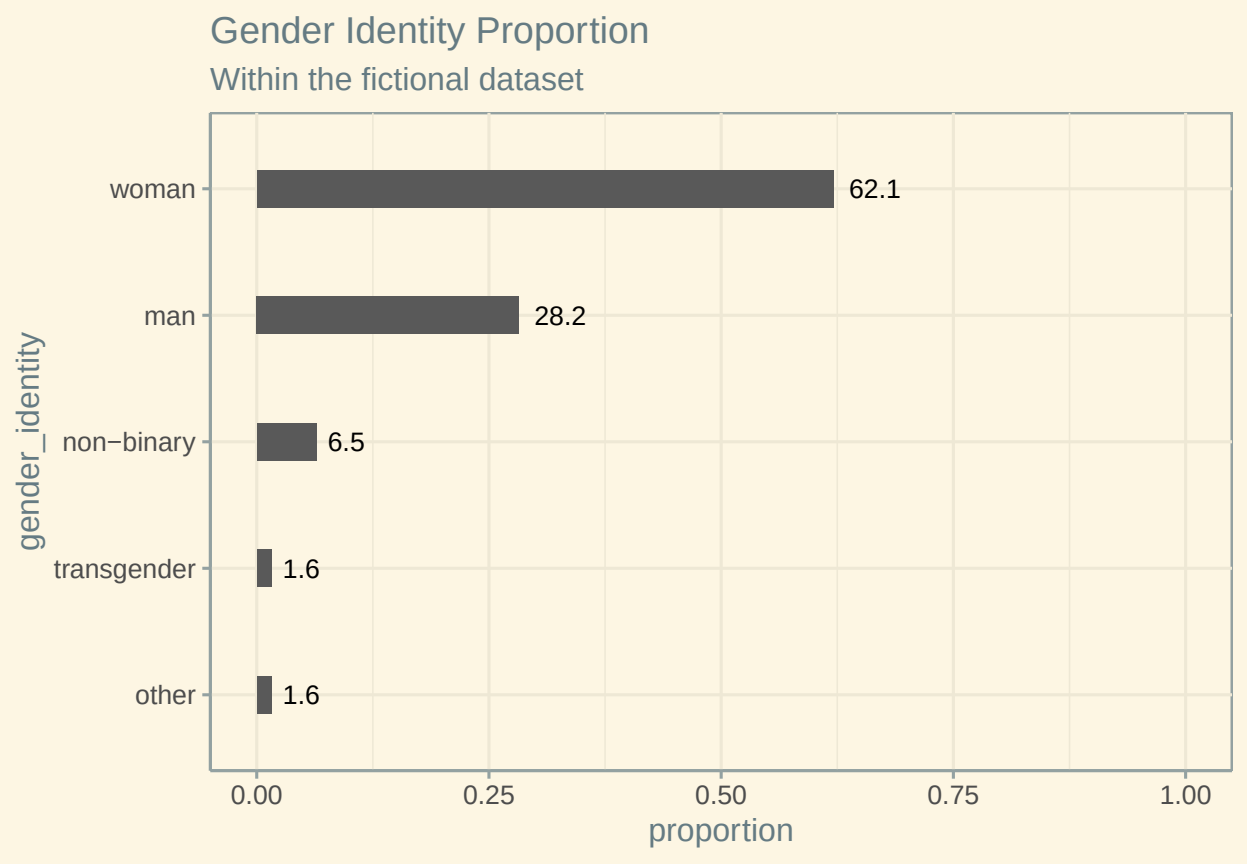

```
plot$lab <- paste( plot$num,
                   paste( " (", plot$perc, "%", sep = "" ),
                   sep = "", ") " )

ggplot(plot, aes( x = gender_identity, y = prop ) ) +
  geom_bar( stat = "identity", width = 0.3 ) +
  coord_flip() +
  ylim( 0 , 1 ) +
  geom_text( aes( x = gender_identity, y = prop, label = lab ),
             size = 3.5, vjust = .5, hjust = -.3 )
```




```
ggplot(plot, aes( x = gender_identity, y = prop ) ) +
  geom_bar( stat = "identity", width = 0.3 ) +
  coord_flip() +
  ylim( 0 , 1 ) +
  geom_text( aes( x = gender_identity, y = prop, label = perc ),
             size = 3.5, vjust = .5, hjust = -.3 ) +
  labs(x = element_blank(),
       y = "proportion",
       title = "Gender Identity Proportion",
       subtitle = "Within the fictional dataset")
```




```
ggplot(plot, aes( x = gender_identity, y = prop ) ) +
  geom_bar( stat = "identity", width = 0.3 ) +
  coord_flip() +
  ylim( 0 , 1 ) +
  geom_text( aes( x = gender_identity, y = prop, label = perc ),
             size = 3.5, vjust = .5, hjust = -.3 ) +
  labs(x = "gender_identity",
       y = "proportion",
       title = "Gender Identity Proportion",
       subtitle = "Within the fictional dataset") +
  theme_solarized()
```




```
ggsave("plots/sample_plot.png")
```

```
## Saving 6.5 x 4.5 in image
```

Part 2:

- scaling axis (values/time)
- hiding x and y elements

For the remainder of the lab, we will use data from the (UC Berkeley Safe Campus study)[https://publichealth.berkeley.edu/covid-19/safe-campus-initiative/charts-and-graphs/] that took place from June to August 2020. The goal of the study was to inform the campus of the safest possible way to re-open campus, if at all.

```
df_covid <- read_csv("data/daily_symptoms.csv") %>% mutate(ts_daily = mdy(ts_daily))
```

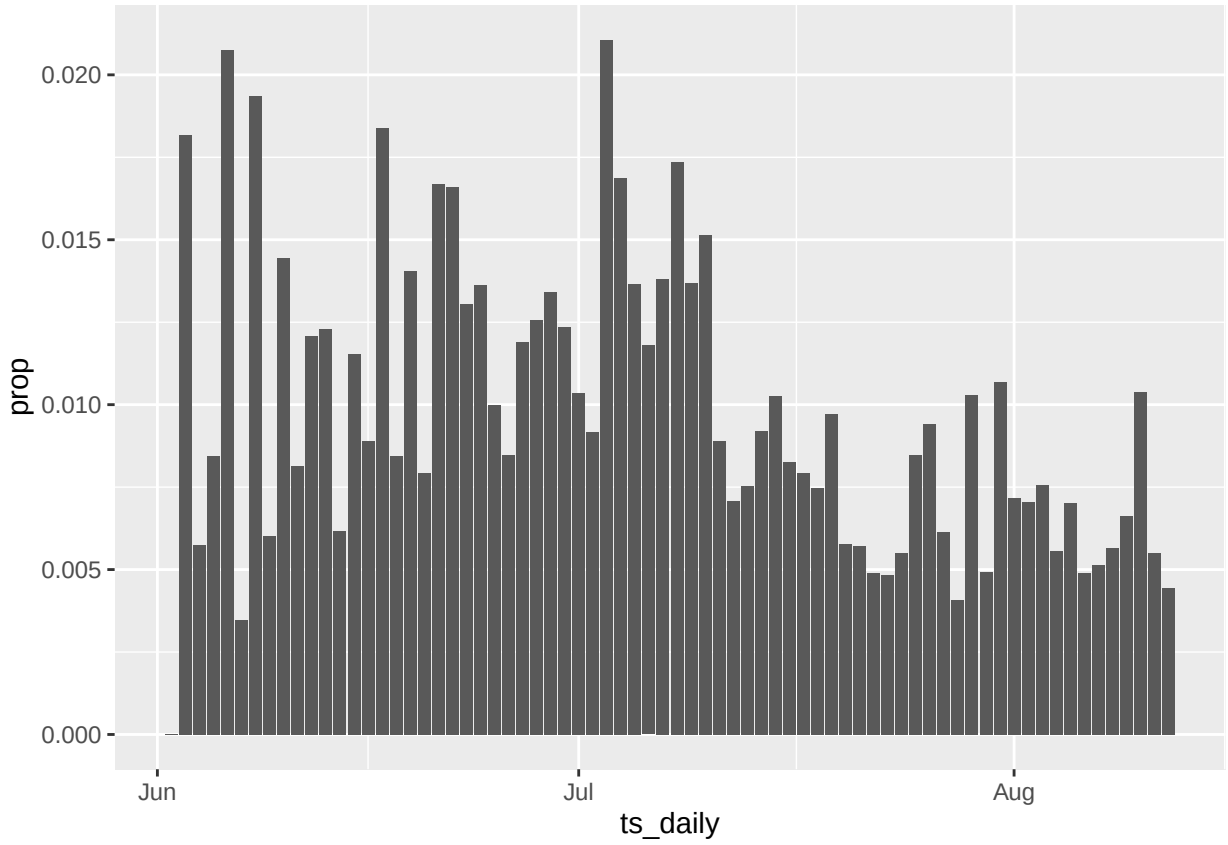
```
## Parsed with column specification:
## cols(
##   ts_daily = col_character(),
##   Blocked_runny_nose = col_double(),
##   Cough = col_double(),
##   Fatigue = col_double(),
##   Feverish = col_double(),
##   Gastrointestinal_symptoms = col_double(),
```

```
##   Loss_of_sense_of_taste_or_smell = col_double(),
##   Muscle_pain_or_body_aches = col_double(),
##   Respiratory_symptoms = col_double(),
##   Sore_throat = col_double()
## )
```

```
symp_1 <- df_covid %>%
  mutate( Blocked_runny_nose = ifelse( Blocked_runny_nose < 0, NA, Blocked_runny_nose )) %>%
  filter(!is.na( Blocked_runny_nose )) %>%
  select( ts_daily, Blocked_runny_nose )

symp_1 <- symp_1 %>%
  group_by( ts_daily ) %>%
  summarize( num = sum(Blocked_runny_nose),
             total = n(),
             prop = num / total ) # proportion per day
```

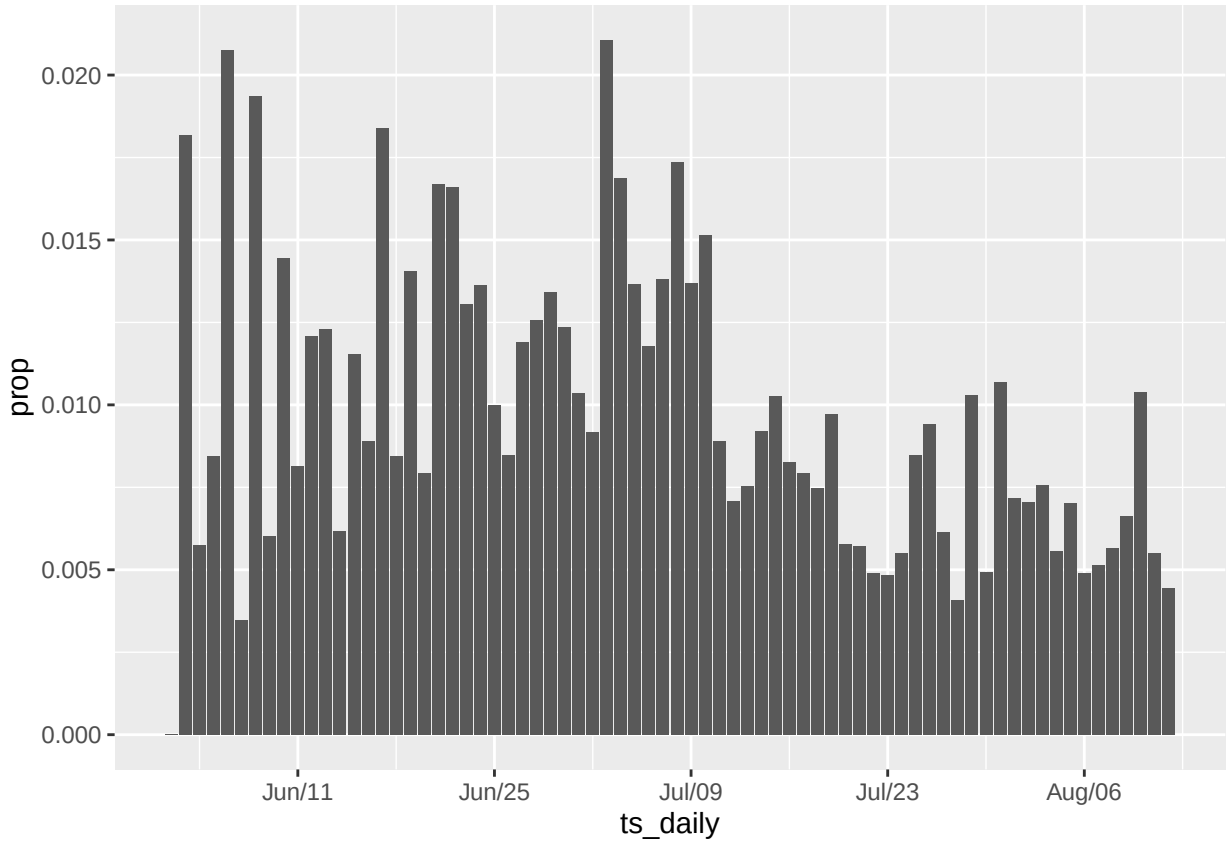
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
ggplot( symp_1, aes( x = ts_daily, y = prop )) +
  geom_col()
```

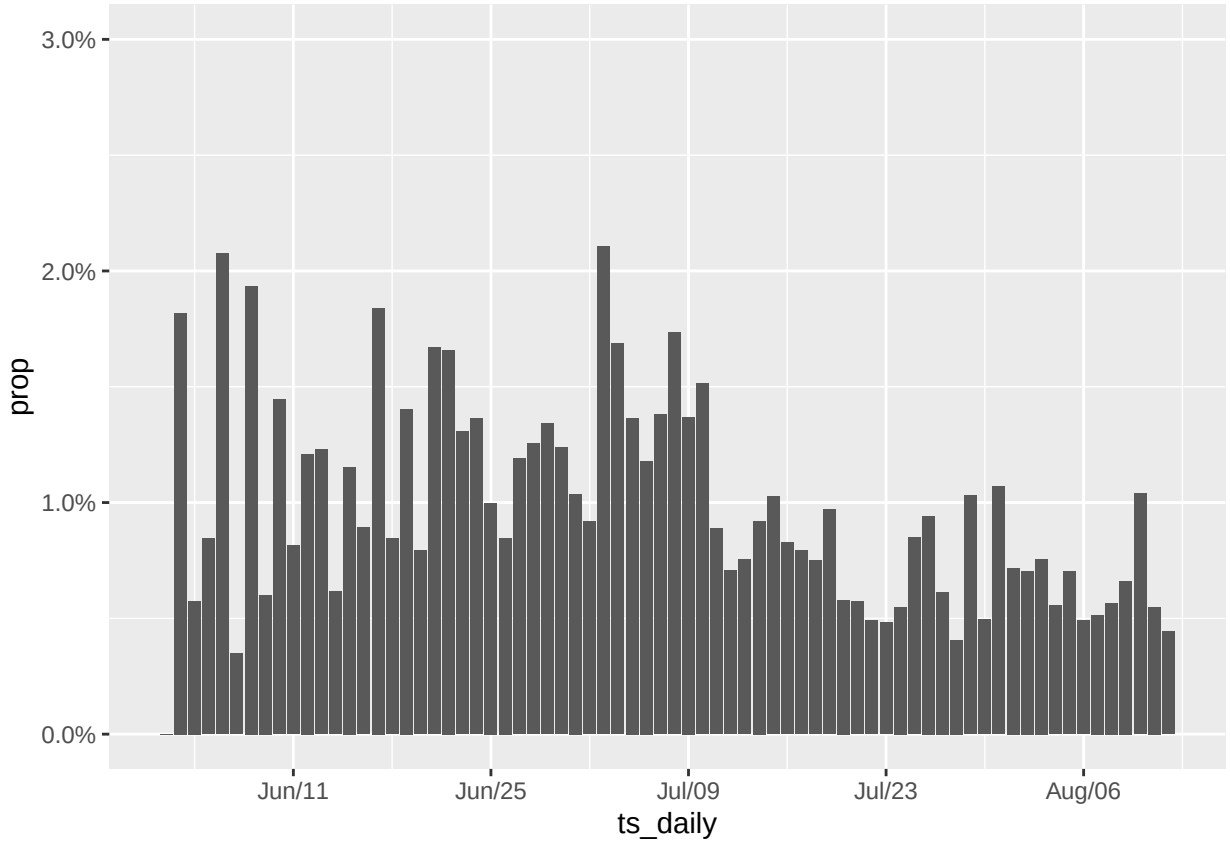



```
# GOOGLE date breaks
```

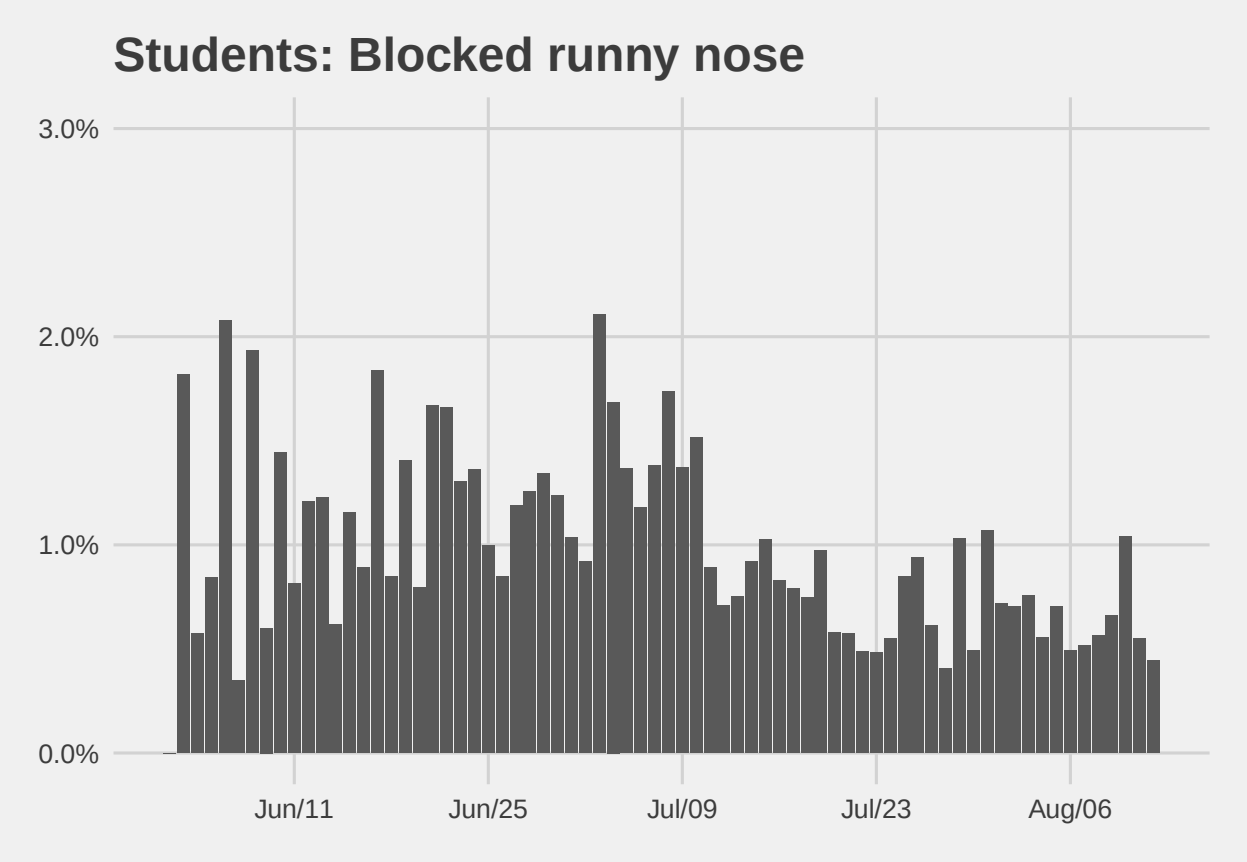
```
ggplot( symp_1, aes( x = ts_daily, y = prop )) +
  geom_col() +
  scale_x_date( date_labels = "%b/%d", date_breaks = "14 days" )
```




```
# GOOGLE y percent axis

ggplot( symp_1, aes( x = ts_daily, y = prop )) +
  geom_col() +
  scale_x_date( date_labels = "%b/%d", date_breaks = "14 days" ) +
  scale_y_continuous( limits = c(0, 0.03), labels = scales::percent)
```




```
ggplot( symp_1, aes( x = ts_daily, y = prop )) +
  geom_col() +
  scale_x_date( date_labels = "%b/%d", date_breaks = "14 days" ) +
  scale_y_continuous( limits = c(0, 0.03), labels = scales::percent) +
  labs(x = element_blank(),
       y = element_blank(),
       title = "Students: Blocked runny nose") +
  theme_fivethirtyeight()
```



Part 3:

- using functions to "automate" graph creation
- detailed modifying of graph element

Wow! That took a lot of time and lines of code, and it was for only one graph! We have 8 more symptoms to go. Let's make a function.

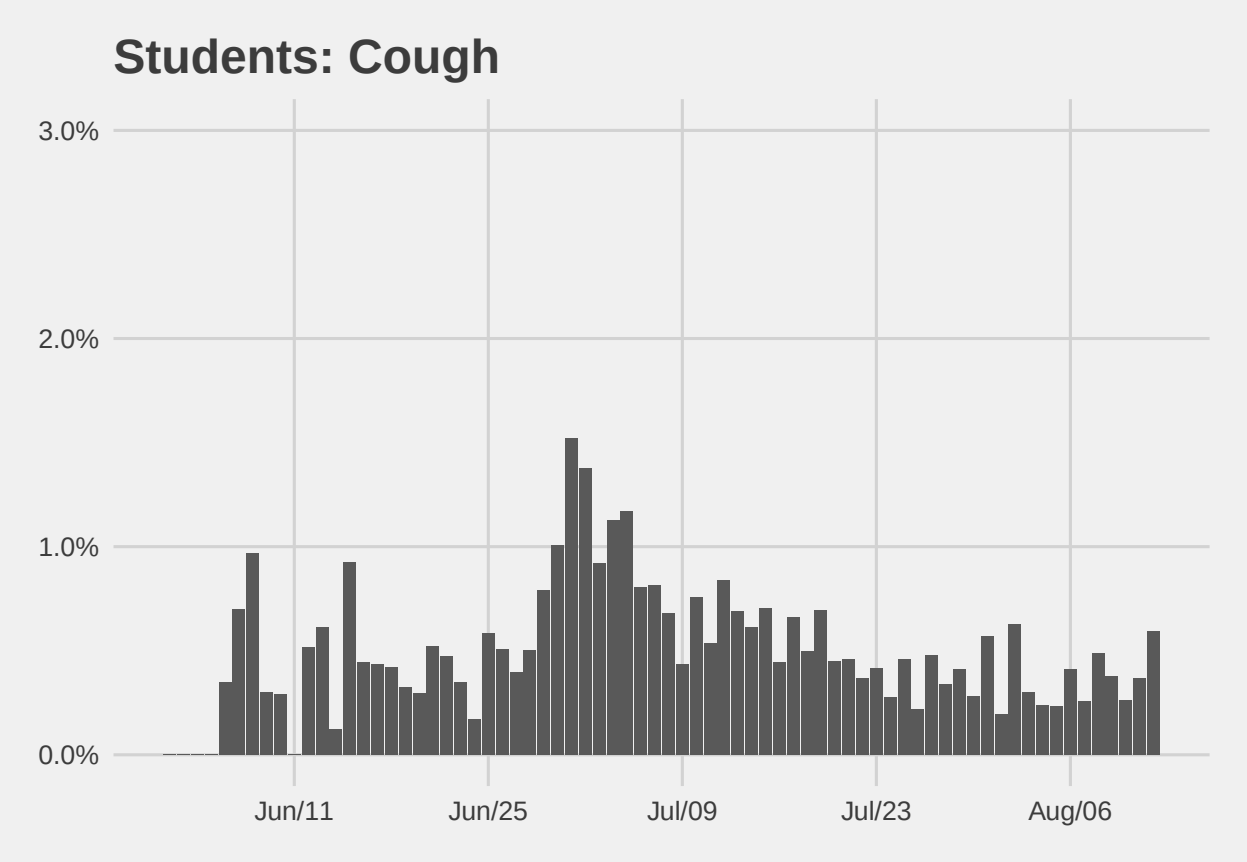
There's a slight variation in the syntax for dyplr when using it within a custom function.

```
symptoms_covid <- function ( data, var ) {
  # create string of variable to later to create column marker
  var.name <- gsub( "_", " ", as.character(substitute( var ) ))
  data %>%
    mutate( var = ifelse( {{ var }} < 0, NA, {{ var }} )) %>%
    # filter out NAs to remove from denominator
    filter( !is.na( var ) ) %>%
    # pair down to variables of interest
    select( ts_daily, var ) %>%
    # group by time
    group_by( ts_daily ) %>%
    summarize( num = sum( var ),          # numerator
               total = n(),               # get total
               prop = num / total ) %>%  # calculate
    # add variable column marker
    mutate(var.name = var.name)
}
```

We can now create the same graph quite quickly for a different symptom:

```
symptoms_covid(df_covid, Cough) %>%
  ggplot( aes(x = ts_daily, y = prop) ) +
    # creating bars
    geom_col() +
    # change x axis to show month/day, separated by 14 days
    scale_x_date( date_labels = "%b/%d", date_breaks = "14 day" ) +
    # change y axis into percent and limit between 0, 0.03
    scale_y_continuous( limits = c(0, 0.03), labels = scales::percent ) +
    # remove x and y titles, and add overall title
    labs(x = element_blank(),
         y = element_blank(),
         title = "Students: Cough") +
    # take advantage of available themes!
    theme_fivethirtyeight()
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```




```
temp_a <- symptoms_covid(df_covid, Blocked_runny_nose)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
temp_b <- symptoms_covid(df_covid, Cough)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
temp_c <- symptoms_covid(df_covid, Fatigue)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
temp_d <- symptoms_covid(df_covid, Feverish)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
temp_e <- symptoms_covid(df_covid, Gastrointestinal_symptoms)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
temp_f <- symptoms_covid(df_covid, Loss_of_sense_of_taste_or_smell)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
temp_g <- symptoms_covid(df_covid, Muscle_pain_or_body_aches)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
temp_h <- symptoms_covid(df_covid, Respiratory_symptoms)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
temp_i <- symptoms_covid(df_covid, Sore_throat)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
symp_plot <- dplyr::bind_rows( temp_a, temp_b, temp_c, temp_d, temp_e,
                               temp_f, temp_g, temp_h, temp_i )



ggplot(symp_plot, aes(x = ts_daily, y = prop, fill = var.name)) +
  geom_col() +
  facet_wrap(~ var.name) +
  scale_x_date(date_labels = "%b/%d", date_breaks = "14 day") +
  scale_y_continuous(limits = c(0, 0.03), labels = scales::percent) +
  labs(x = element_blank(),
       y = element_blank(),
       title = "Daily student reported symptoms",
       subtitle = "by type of symptom",
       caption = "for entire duration of the study",
```

```
      color = element_blank()) +
  theme_fivethirtyeight() +
  theme(plot.title = element_text(color = "#003262"),
        plot.subtitle = element_text(color = "#FDB515"),
        panel.grid.major.x = element_blank(),
        plot.background = element_rect(fill = "white", colour = "white"),
        panel.background = element_rect(fill = "white", color = "white"),
        strip.background = element_rect(colour = "white", fill = "white"),
        legend.position = "none",
        axis.text.x = element_text(angle= 45, hjust = 1),
        axis.text.y = element_text()) +
  guides(colour = guide_legend(nrow = 3))
```

```
## Warning: Removed 3 rows containing missing values (position_stack).
```