

TidyTuesday: Colour palettes and volcanoes

Liza Bolton

2020-05-12

- This session will be recorded and put up on Past events
- Click the stacked lines at the top left of this panel to open a helpful navigation pane

Code for custom colour palettes

You can see this in action at

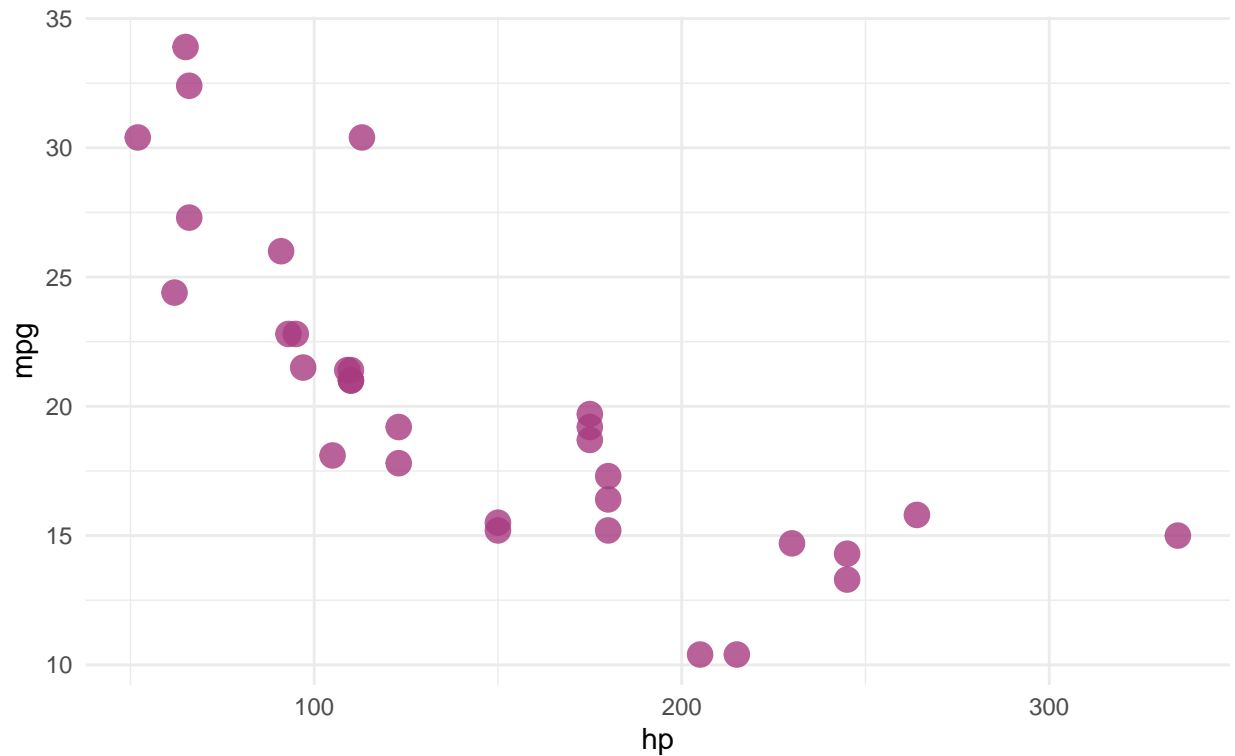
```
library(ggplot2)
# Set up the colours you want
my_colours <- c(
  `turquoise` = "#30E6C9",
  `khaki`     = "#D7EA85",
  `slate`     = "#355E78",
  `old-rose`  = "#A83B80",
  `cadet-blue` = "#77A5BF",
  `light grey` = "#cccccc", # added extra neutrals
  `dark grey` = "#8c8c8c")

#Function that converts names to hexcodes
get_spiderverse_col <- function(...) {
  cols <- c(...)

  if (is.null(cols)) return (my_colours)

  my_colours[cols]
}

# You could just use this function:
ggplot(mtcars, aes(hp, mpg)) +
  geom_point(color = get_spiderverse_col("old-rose"),
    size = 4, alpha = .8) +
  theme_minimal()
```



```
].pull-right[
]
```

Setting up your own palette: Set up scales for ggplot

```
# Colour scale - good for scatter plots
scale_color_spiderverse <- function(palette = "main", discrete = TRUE, reverse = FALSE, ...) {
  pal <- spiderverse_pal(palette = palette, reverse = reverse)

  if (discrete) {
    discrete_scale("colour", paste0("spiderverse_", palette), palette = pal, ...)
  } else {
    scale_color_gradientn(colours = pal(256), ...)
  }
}

# Fill scale - good for barplots
scale_fill_spiderverse <- function(palette = "main", discrete = TRUE, reverse = FALSE, ...) {
  pal <- spiderverse_pal(palette = palette, reverse = reverse)

  if (discrete) {
    discrete_scale("fill", paste0("spiderverse_", palette), palette = pal, ...)
  } else {
    scale_fill_gradientn(colours = pal(256), ...)
  }
}
```

Set up packages

- I always love using `tidyverse` (which includes `ggplot`, `dplyr` and several other helpful packages).
- The `devtools` (developer tools) package lets me download packages that aren't on CRAN.
- `emo` makes it easy to use emoji

```
list.of.packages <- c("tidyverse", "devtools", "leaflet",
                     "emo", "yarr", "ggthemes")
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages, repos = "http://cran.us.r-project.org")

rm("list.of.packages", "new.packages")

library(tidyverse)
```

```
## -- Attaching packages ----- tidyverse 1.3.0 --
```

```
## v tibble  3.0.1      v dplyr    0.8.5
## v tidyr   1.0.0      v stringr 1.4.0
## v readr   1.3.1      v forcats 0.5.0
## v purrr   0.3.4
```

```
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()     masks stats::lag()
```

```
library(leaflet)
library(ggthemes)
# devtools::install_github("ropenscilabs/ochRe")
library(ochRe)
# devtools::install_github("ropenscilabs/icon")
library(icon)
```

Load the data

At the top of every tidy Tuesday there are a range of ways you can get the data. Last week I just copied the links that use `readr::read_csv()` to read the files from their URLs. This week I'm using the `tidytuesdayR` package.

```
#####
# NOTE: install this to use tidytuesdayR functions
# devtools::install_github("thebioengineer/tidytuesdayR")
#####

tuesdata <- tidytuesdayR::tt_load('2020-05-12')
```

```
## --- Downloading #TidyTuesday Information for 2020-05-12 ----
```

```
## --- Identified 5 files available for download ----
```

```
## --- Downloading files ---
```

```
## --- Download complete ---
```

```
tuesdata
```

```
## Available datasets:
```

```
## volcano
```

```
## eruptions
```

```
## events
```

```
## tree_rings
```

```
## sulfur
```

```
##
```

```
# The below would do the same thing
```

```
# tuesdata <- tidyTuesdayR::tt_load(2020, week = 20)
```

```
volcano <- tuesdata$volcano
```

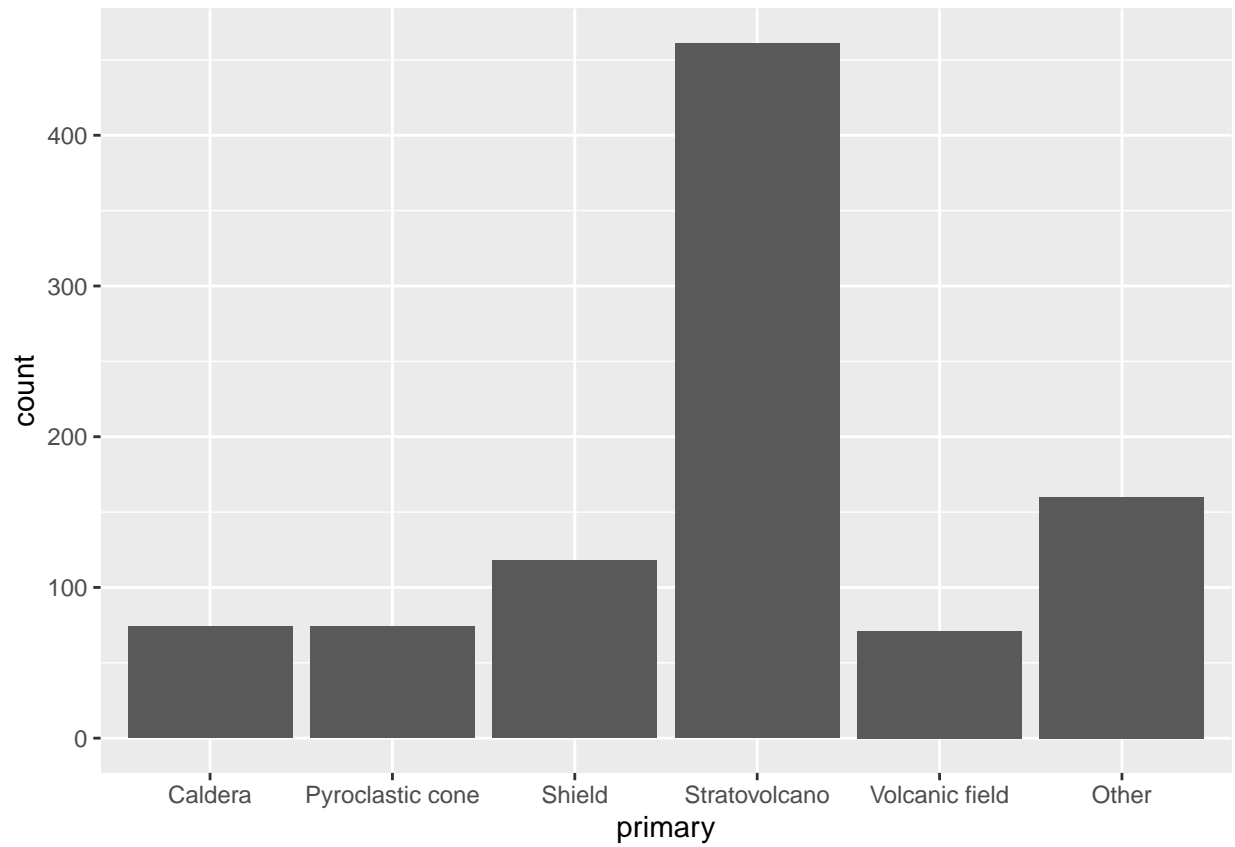
Let's explore

```
volcano_edit <- volcano %>%  
  mutate(last_eruption_year = parse_number(last_eruption_year)) %>%  
  mutate(primary = gsub("\\s*\\([^\\)]+\\)", "", primary_volcano_type)) %>%  
  mutate(primary = gsub("\\\\?", "", primary)) %>%  
  mutate(primary = fct_lump_n(primary, 5))
```

```
## Warning: 301 parsing failures.  
## row col expected actual  
## 2 -- a number Unknown  
## 7 -- a number Unknown  
## 8 -- a number Unknown  
## 10 -- a number Unknown  
## 17 -- a number Unknown  
## ... ..  
## See problems(...) for more details.
```

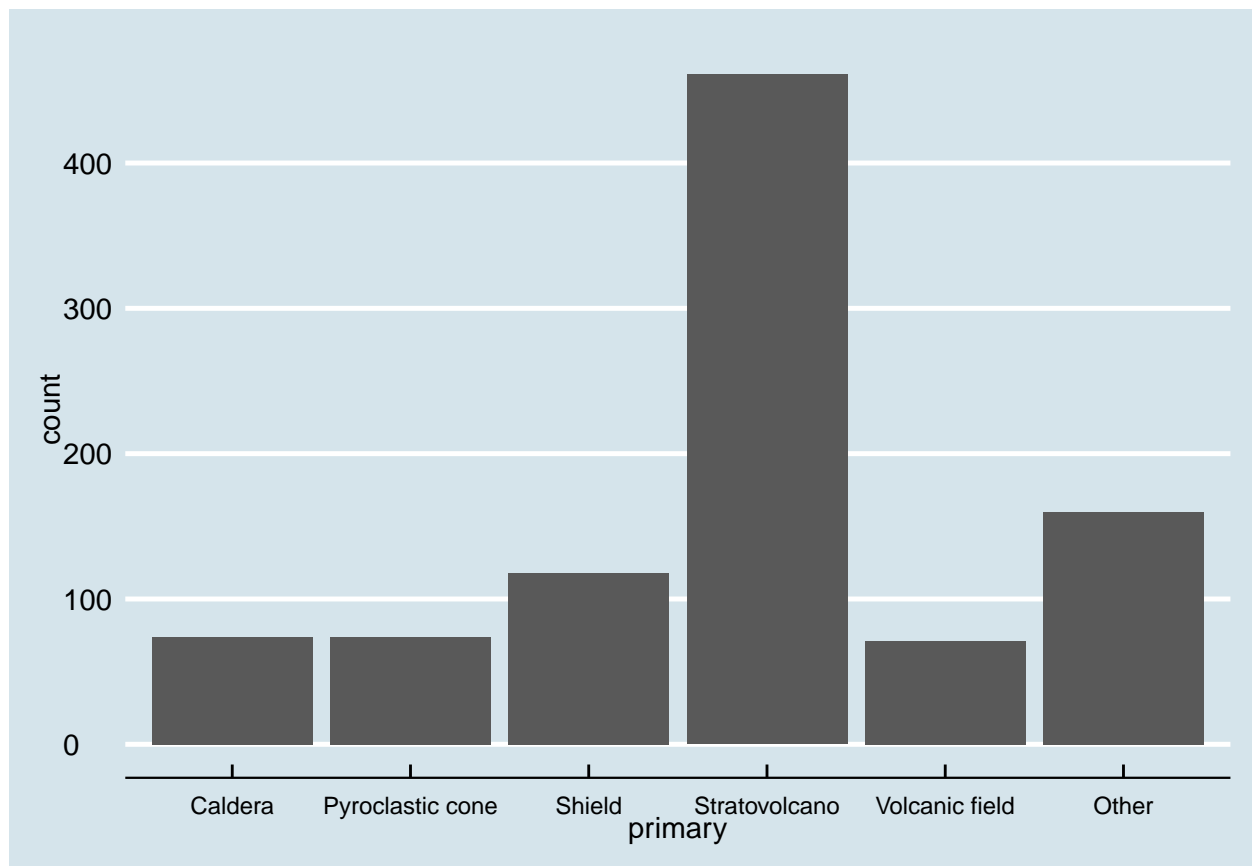
```
primary <- volcano_edit %>%  
  ggplot(aes(x = primary)) +  
  geom_bar()
```

```
primary
```



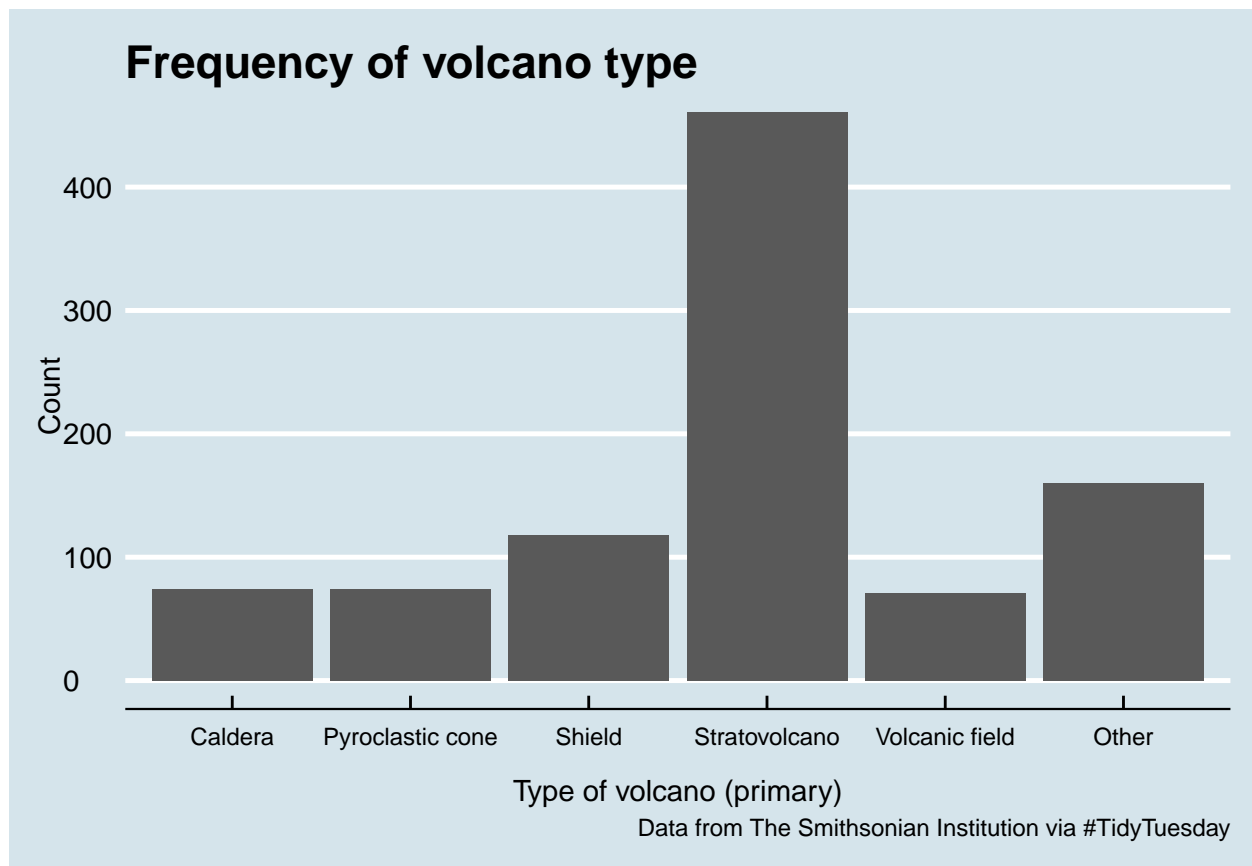
```
primary <- primary +  
  theme_economist() +  
  theme(axis.text.x = element_text(size = 9))
```

```
primary
```



```
primary <- primary +  
  xlab("\nType of volcano (primary)") +  
  ylab("Count") +  
  labs(title = "Frequency of volcano type",  
        caption = "Data from The Smithsonian Institution via #TidyTuesday")
```

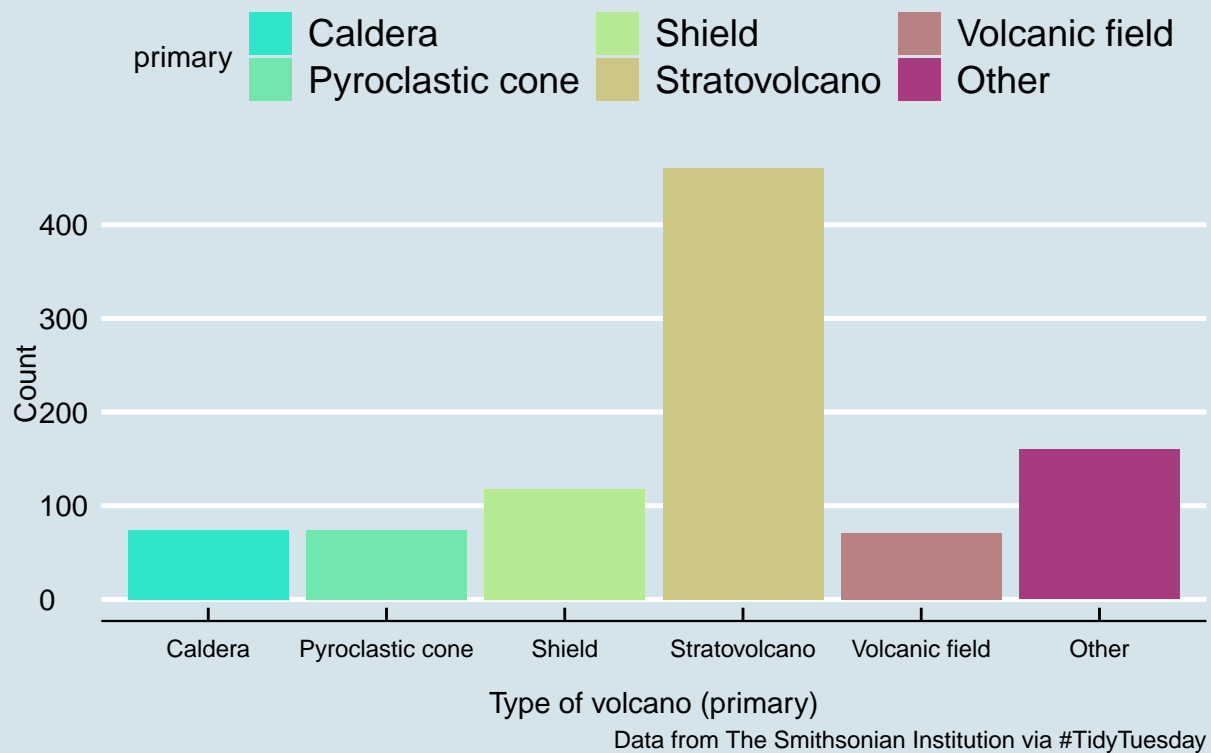
```
primary
```



```
primary <- primary +  
  geom_bar(aes(fill=primary)) +  
  scale_fill_spiderverse("main")
```

```
primary
```

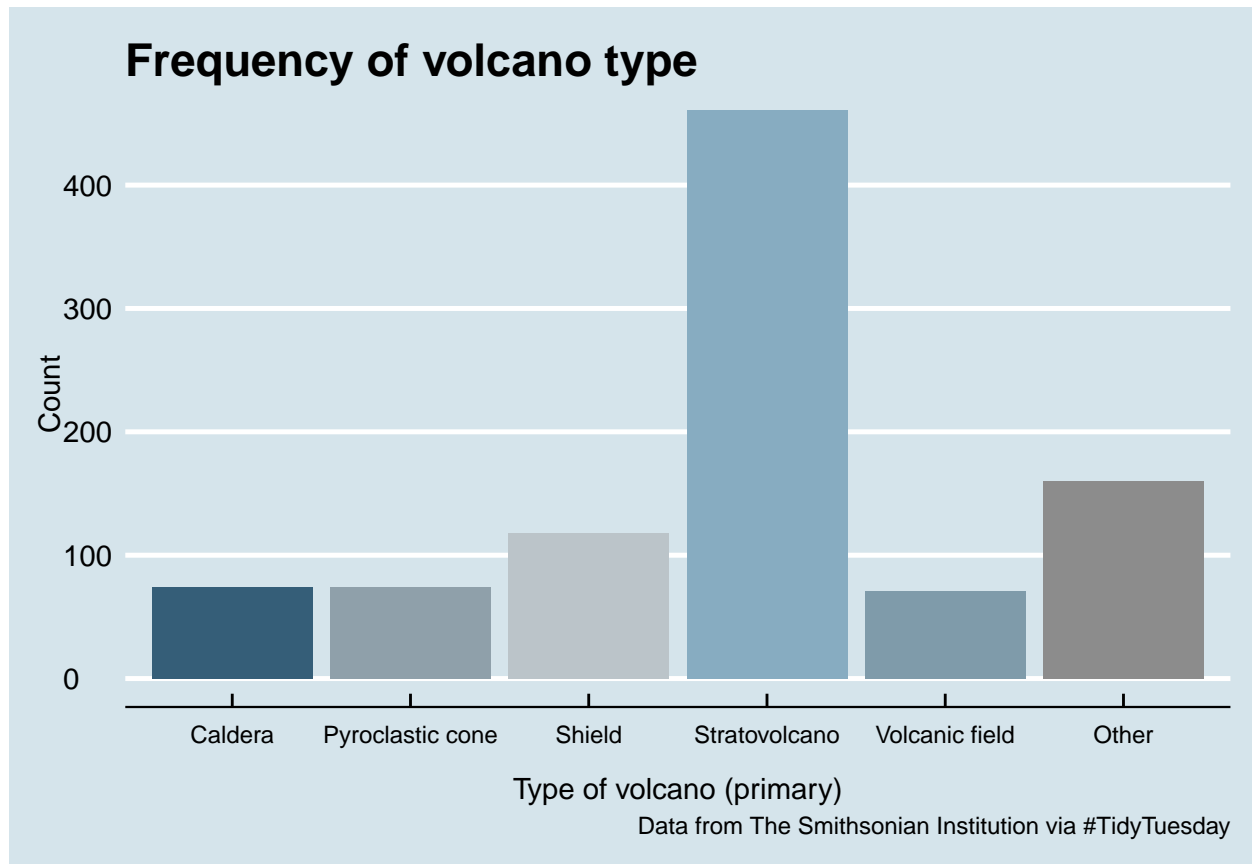
Frequency of volcano type



```
primary <- primary +  
  geom_bar(aes(fill=primary)) +  
  scale_fill_spiderverse("neutral") +  
  theme(legend.position = "none")
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which  
## will replace the existing scale.
```

```
primary
```

```
# Leaflet plot
volcano_edit %>%
  # Let's focus on the last 20 years
  filter(last_eruption_year > 2000) %>%
  leaflet() %>%
  # A nice minimal map
  addTiles() %>%
  addAwesomeMarkers(
    lat = ~latitude,
    lng = ~longitude,
    #volcano shaped markers
    icon = awesomeIcons(
      library = "fa",
      icon = "caret-up",
      iconColor = "red",
      markerColor = "cadetblue"),
    label = ~paste0(volcano_name, ", ", last_eruption_year, ", ", primary))
```

