SIMILARITY MEASURE IN THE SPACE OF

COLOR PALETTES

Color exploration with
Emil Hvitfeldt Hansen
Twitter: @Emil_Hvitfeldt
Github: EmilHvitfeldt

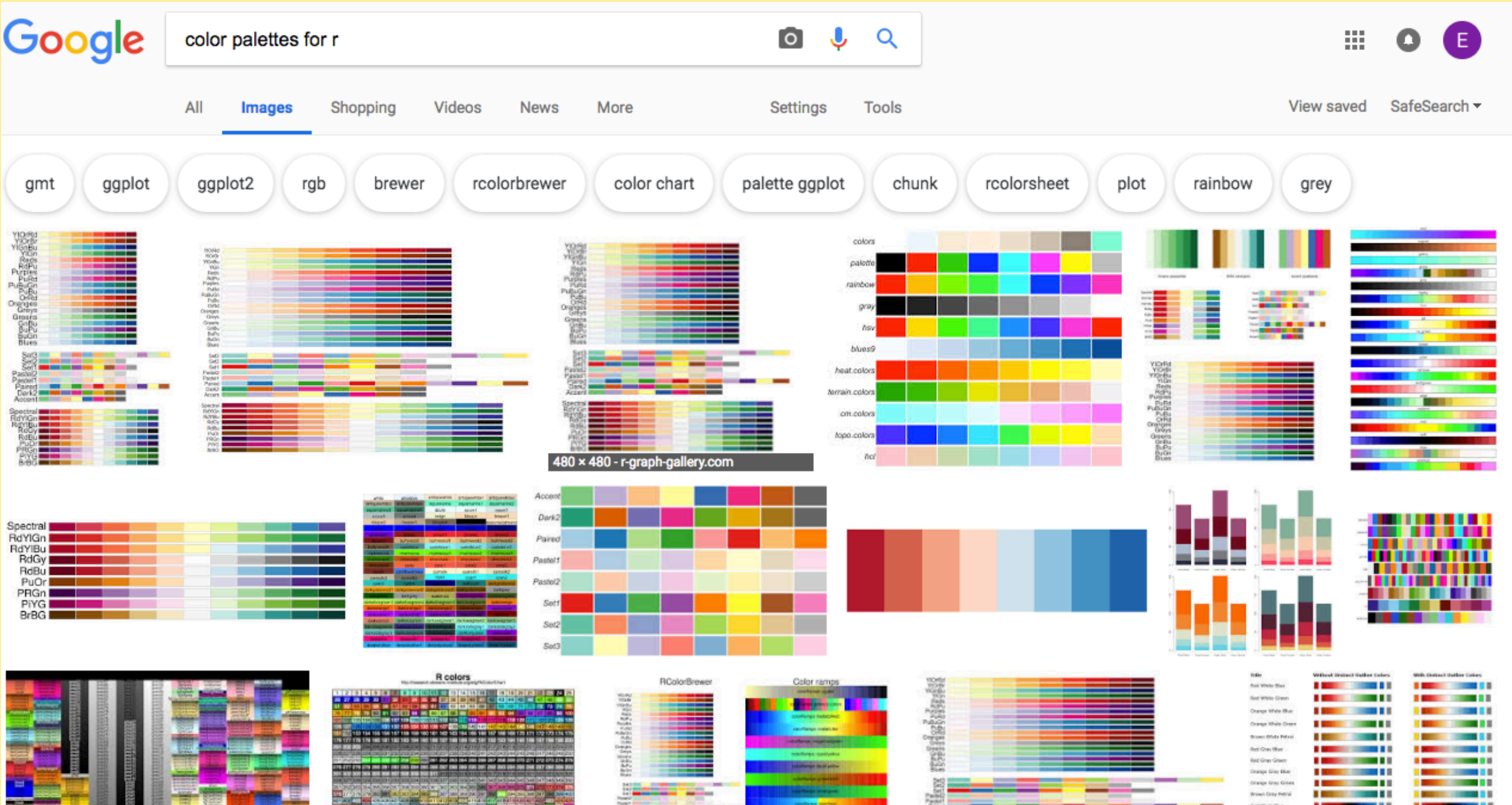
github.com/EmilHvitfeldt/OCRUG-color-talk
bit.ly/OCRUG-Color-Talk

OVERVIEW

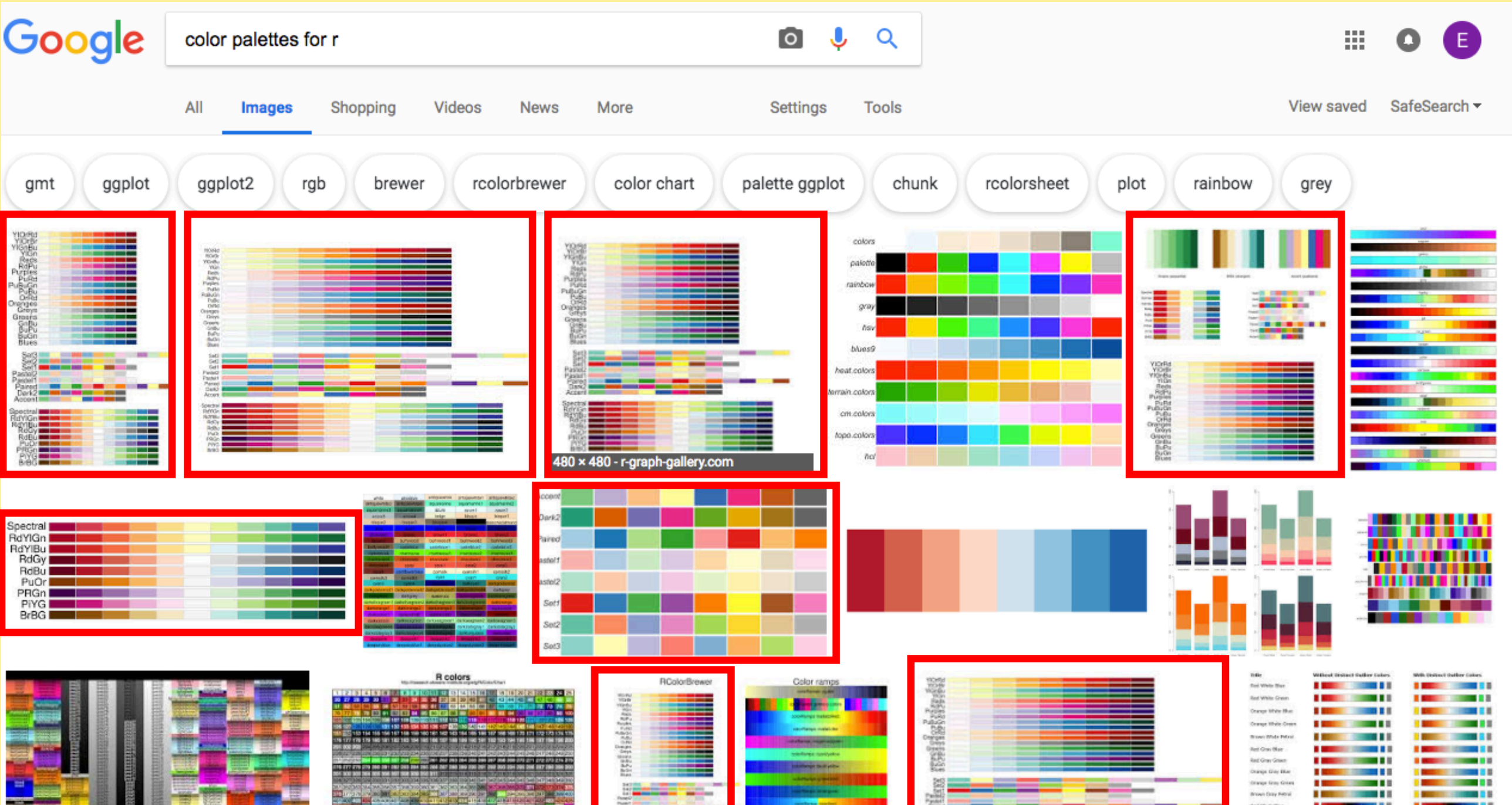
- Overview
- Motivation
- Method
- Results

AVAILABLE COLOR PALETTES

AVAILABLE COLOR PALETTES

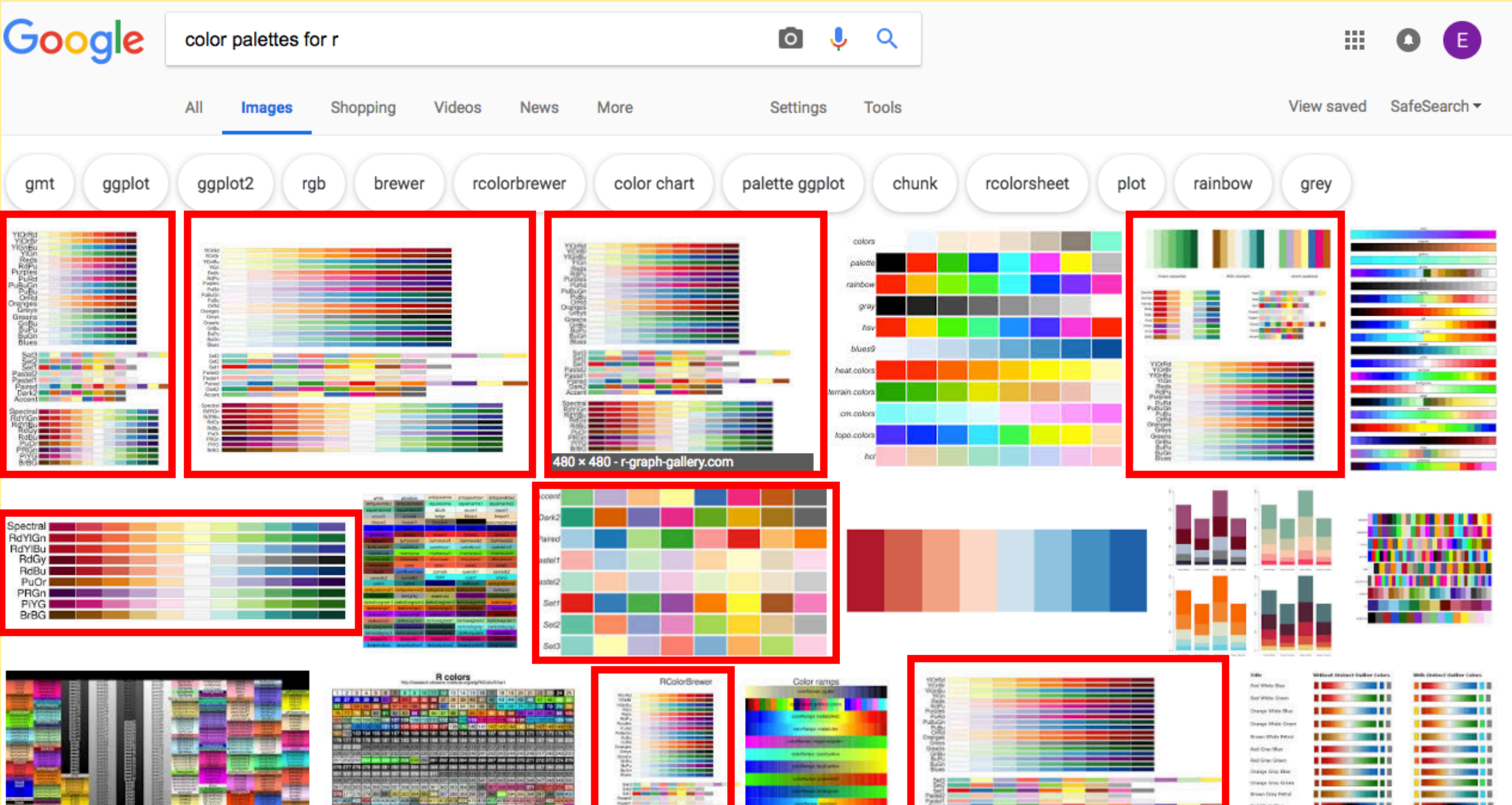


AVAILABLE COLOR PALETTES



RColorBrewer

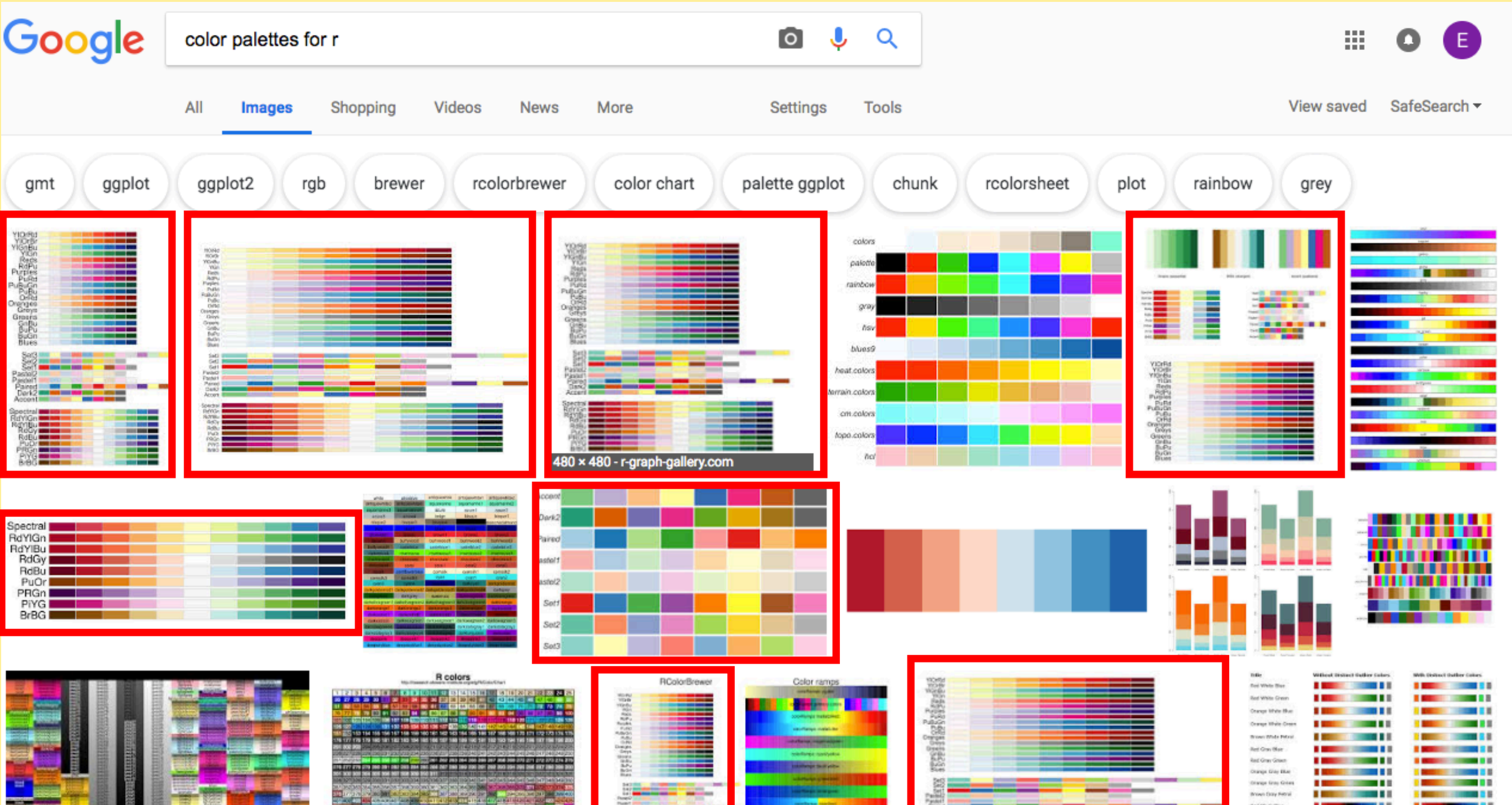
AVAILABLE COLOR PALETTES



RColorBrewer

AVAILABLE COLOR PALETTES

RColorBrewer

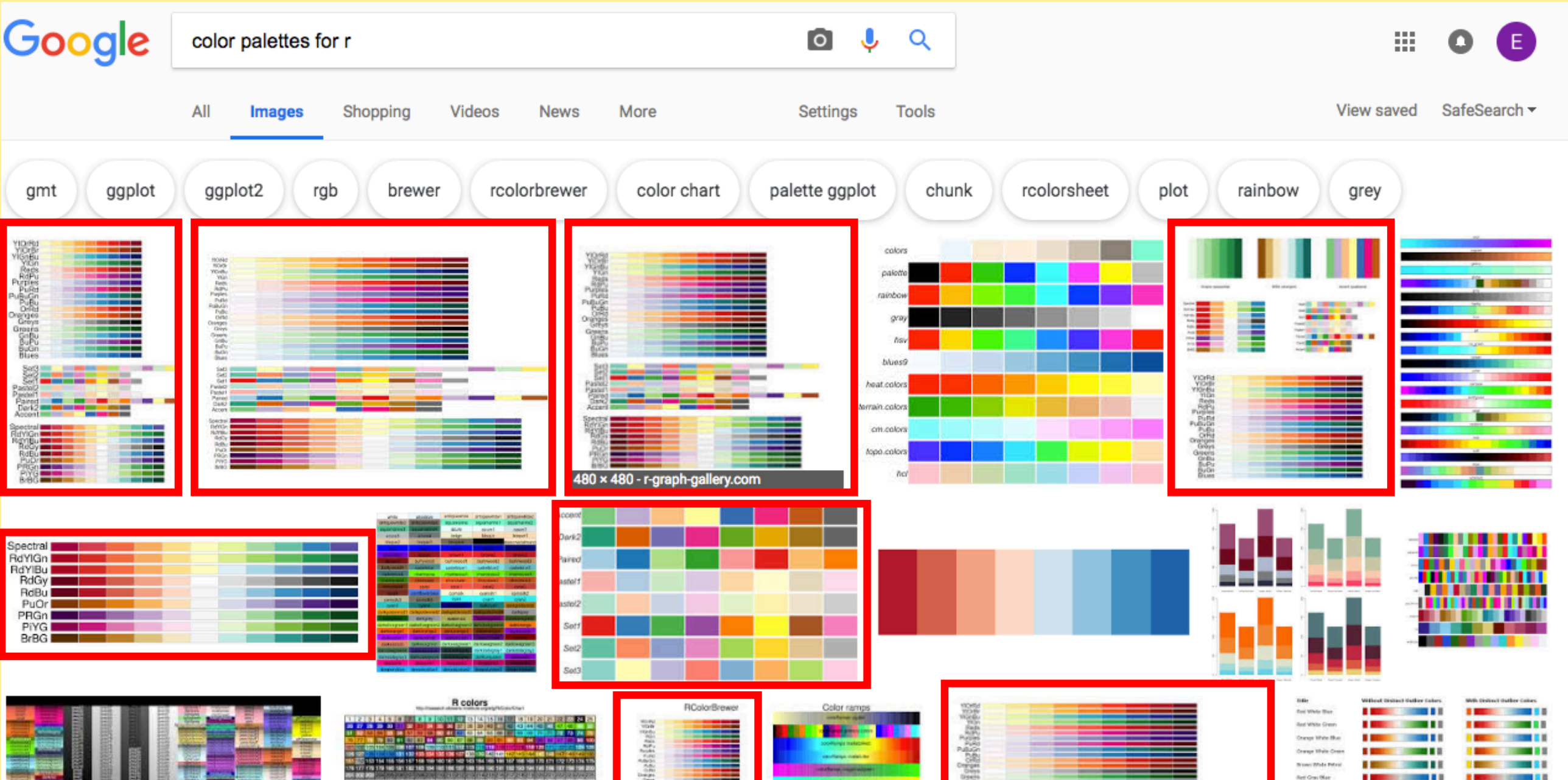


RColorBrewer

AVAILABLE COLOR PALETTES

RColorBrewer

RColorBrewer

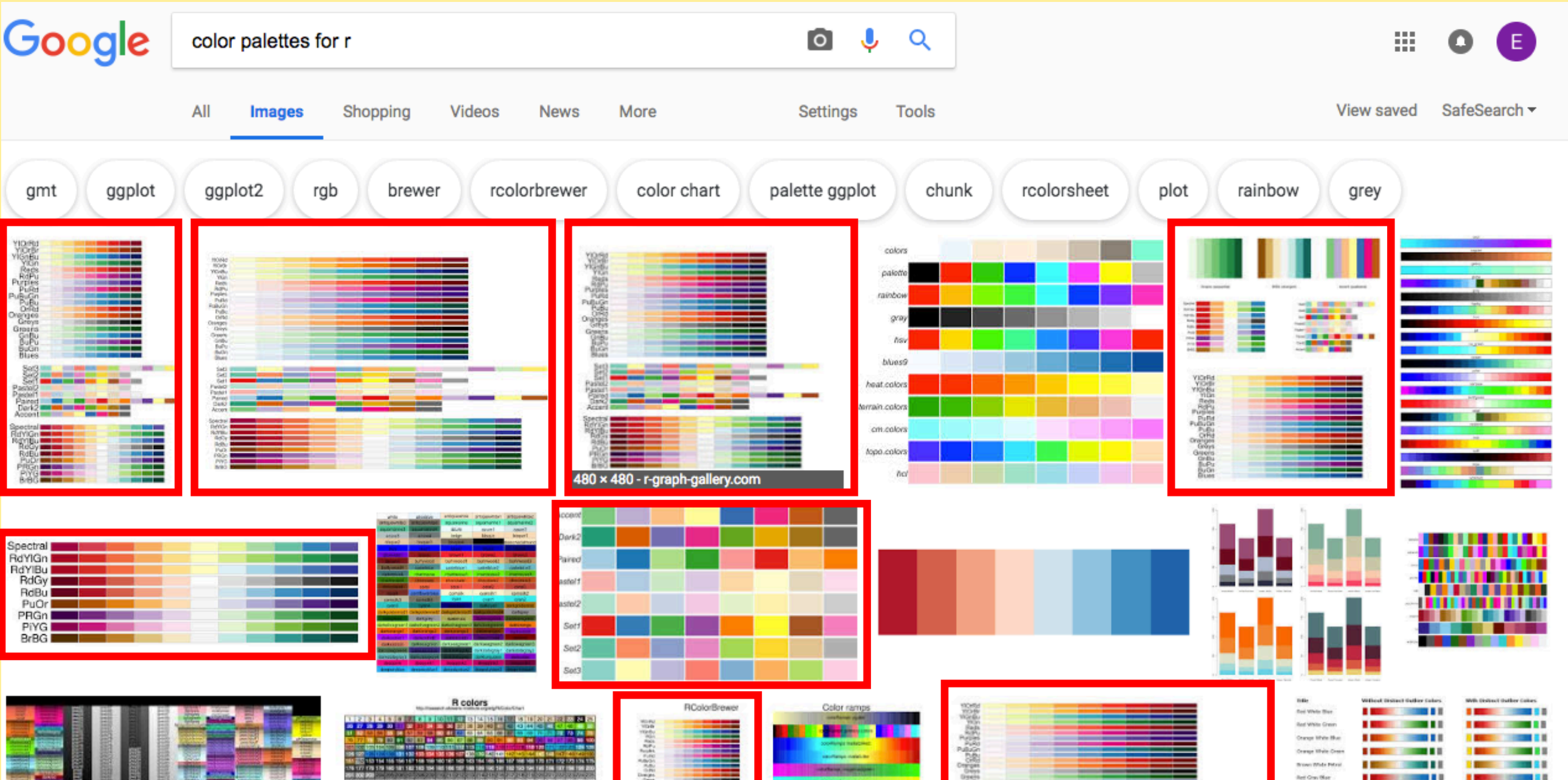


RColorBrewer RColorBrewer

AVAILABLE COLOR PALETTES

RColorBrewer

RColorBrewer

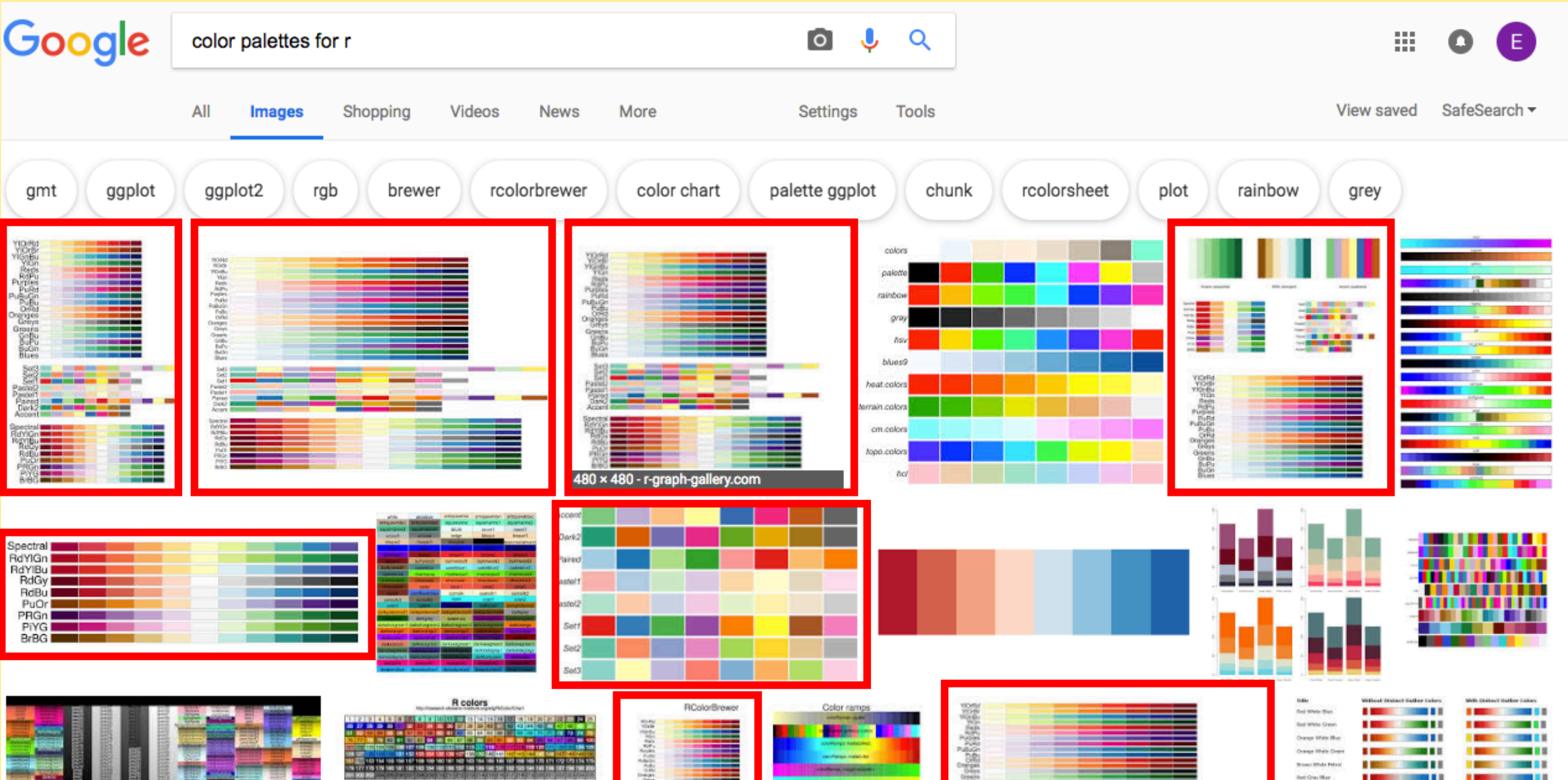


RColorBrewer RColorBrewer

AVAILABLE COLOR PALETTES

RColorBrewer

RColorBrewer RColorBrewer



SOMETHING IS MISSING

- Where is the **viridis palettes**?
- Where is the **scico palettes**?
- Where is the **nord palettes**?
- Where is the **ochRe palettes**?
- …

EmilHvitfeldt / r-color-palettes

Unwatch 17 | Unstar 246 | Fork 24

Code | Issues 2 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

Comprehensive list of color palettes available in r

Edit

rstats palette palettes palette-generation color color-palette r Manage topics

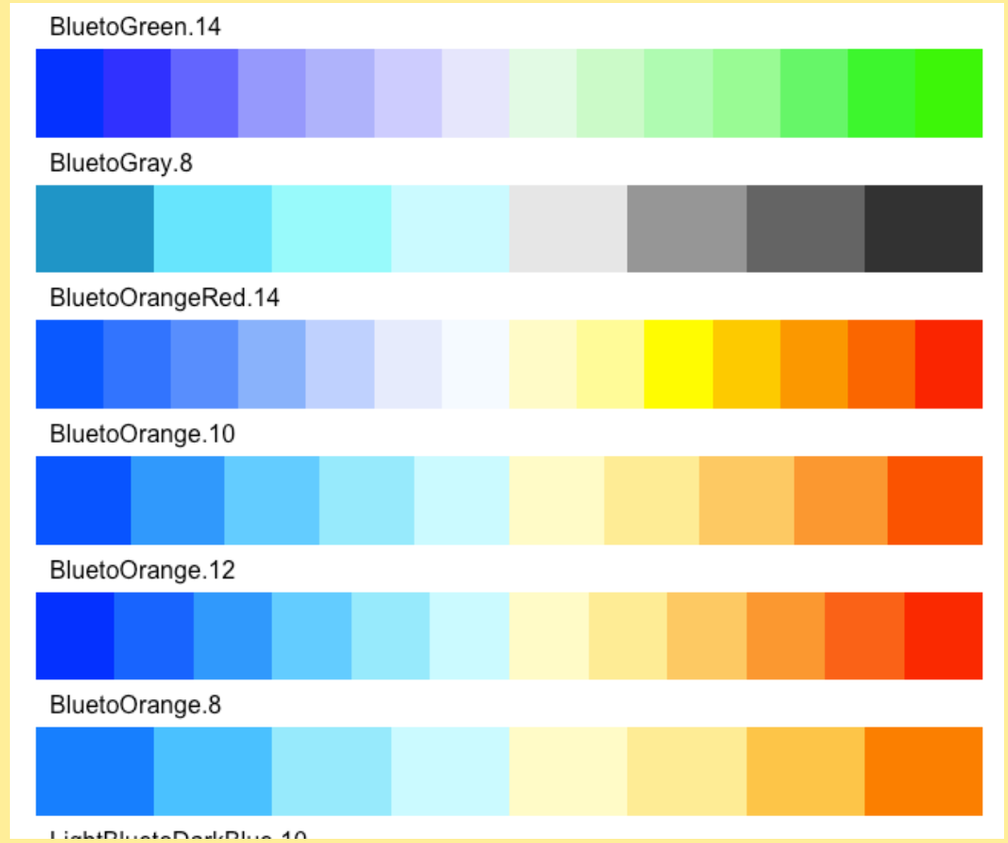
50 commits | 1 branch | 0 releases | 4 contributors

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

EmilHvitfeldt redone type sorted palettes — Latest commit d6a319c on Jul 18

R	redone type sorted palettes	2 months ago
man/figures	redone type sorted palettes	2 months ago
.Rbuildignore	Initial release, palettes alphabetically	7 months ago
.gitignore	adding full palettetown palette on seperate page. closes #31.	2 months ago
NEWS.md	redone type sorted palettes	2 months ago
README.Rmd	finish readme with new style	2 months ago
README.md	finish readme with new style	2 months ago
canva.Rmd	adding full palettetown palette on seperate page. closes #31.	2 months ago
palettetown.Rmd	adding full palettetown palette on seperate page. closes #31.	2 months ago
palettetown.md	adding full palettetown palette on seperate page. closes #31.	2 months ago
r-color-palettes.Rproj	Initial release, palettes alphabetically	7 months ago
type-sorted-palettes.Rmd	redone type sorted palettes	2 months ago
type-sorted-palettes.md	redone type sorted palettes	2 months ago

958 palettes from 28 packages



paletteer

The goal of **paletteer** is to be a comprehensive collection of color palettes in R using a common interface. Think of it as the "caret of palettes".

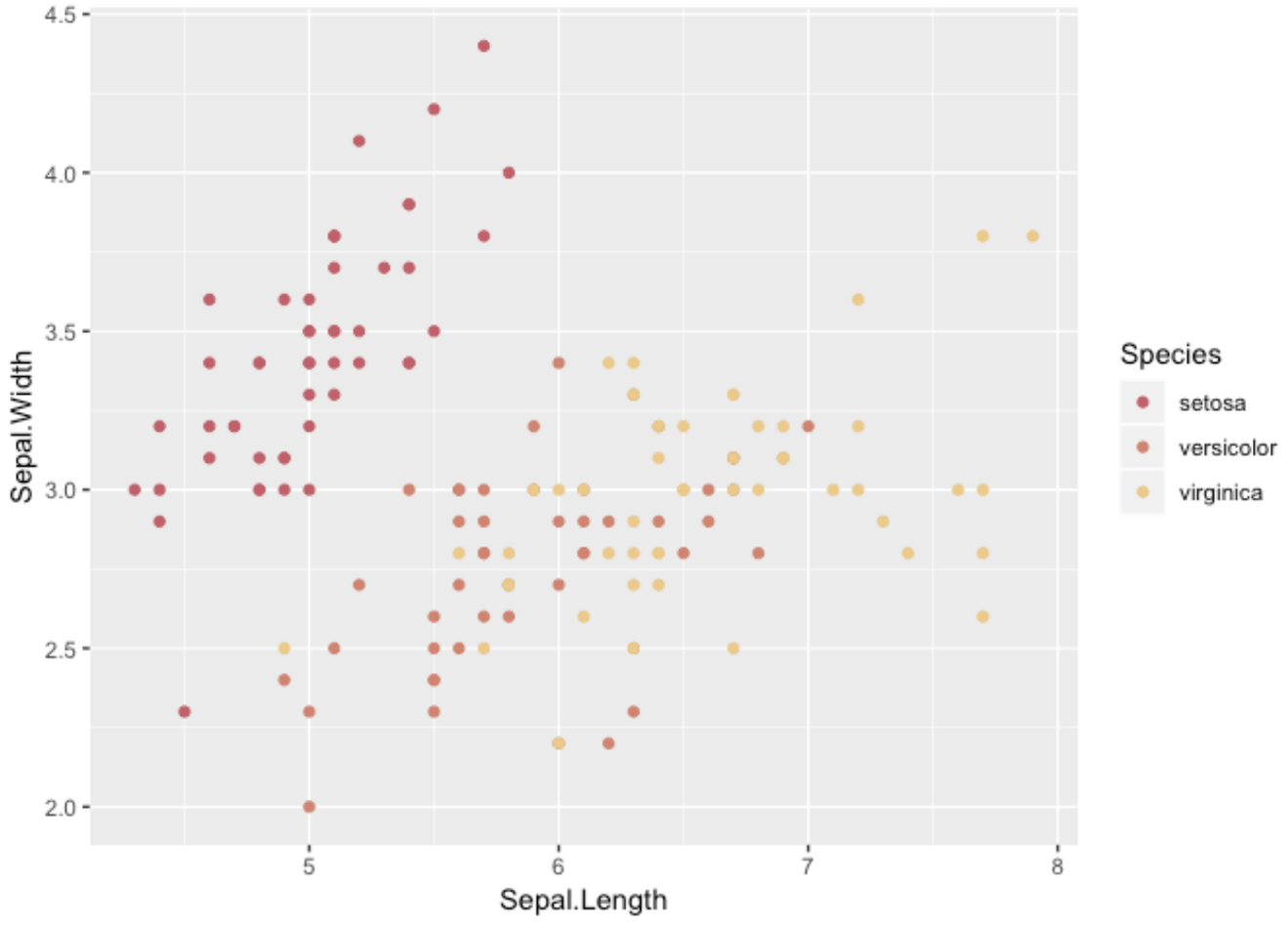
```
paletteer_c(package = "scico", palette = "berlin", n = 10)
#>  [1] "#9EB0FF" "#5AA3DA" "#2D7597" "#194155" "#11181D" "#270C01" "#501802"
#>  [8] "#8A3F2A" "#C37469" "#FFACAC"
paletteer_d("nord", "frost")
#> [1] "#8FBCBB" "#88C0D0" "#81A1C1" "#5E81AC"
paletteer_dynamic("cartography", "green.pal", 5)
#> [1] "#B8D9A9" "#8DBC80" "#5D9D52" "#287A22" "#17692C"
```

```
paletteer_c(scico, berlin, 10)
#>  [1] "#9EB0FF" "#5AA3DA" "#2D7597" "#194155" "#11181D" "#270C01" "#501802"
#>  [8] "#8A3F2A" "#C37469" "#FFACAC"
paletteer_d(nord, frost)
#> [1] "#8FBCBB" "#88C0D0" "#81A1C1" "#5E81AC"
paletteer_dynamic(cartography, green.pal, 5)
#> [1] "#B8D9A9" "#8DBC80" "#5D9D52" "#287A22" "#17692C"
```

paletteer

The goal of **paletteer** is to be a comprehensive collection of color palettes in R using a common interface. Think of it as the "caret of palettes".

```
library(ggplot2)

ggplot(iris, aes(Sepal.Length, Sepal.Width, color = Species)) +
  geom_point() +
  scale_color_paletteer_d(nord, aurora)
```



MY JOB IS DONE?

MY JOB IS DONE?

MY JOB IS DONE?

- New problem with discoverability

MY JOB IS DONE?

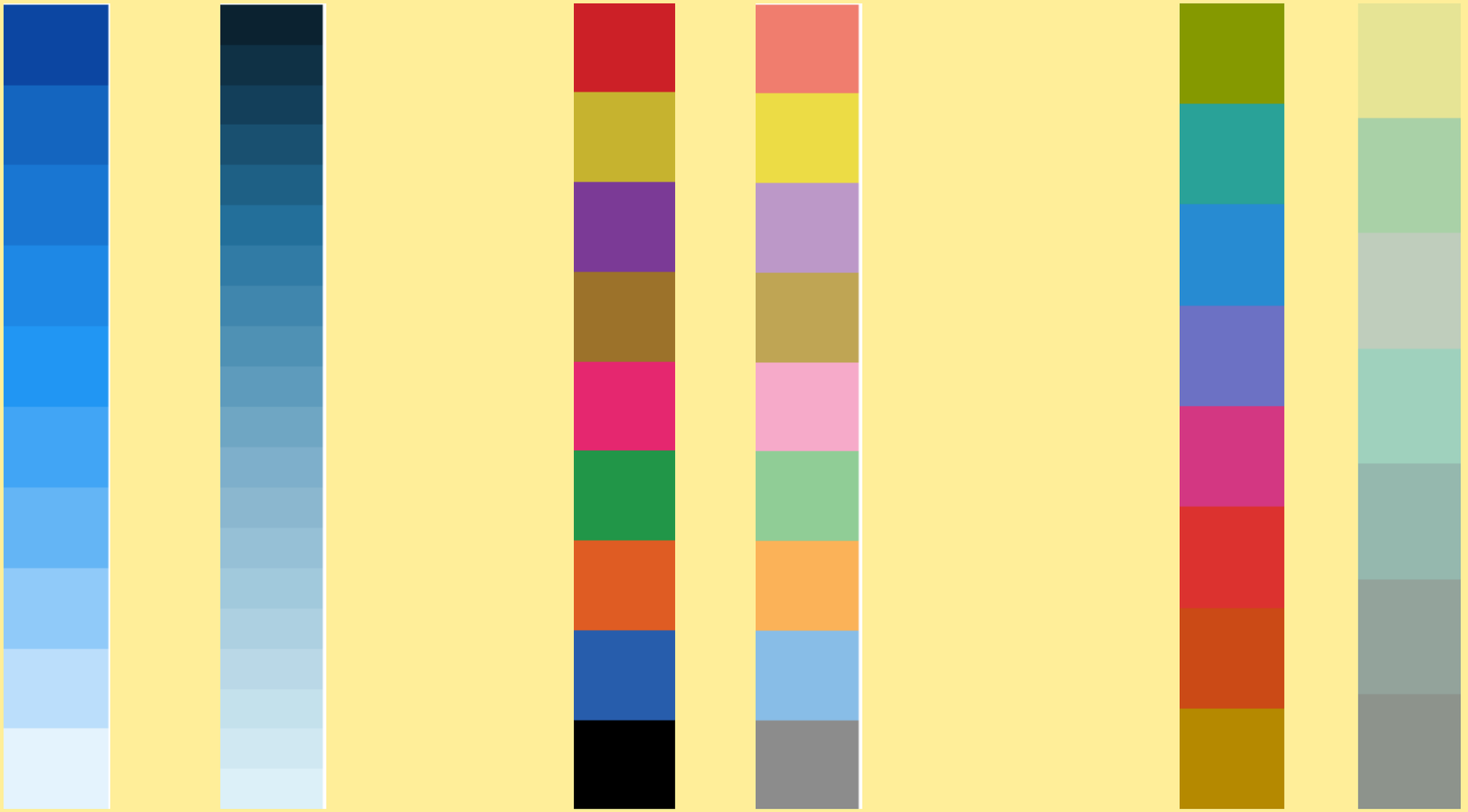
- New problem with discoverability
- 100s of palettes, hard to find a new one without using r-color-palettes

MY JOB IS DONE?

- New problem with discoverability
- 100s of palettes, hard to find a new one without using r-color-palettes
- Still using same palettes

PLAN

- Develop tool to find palette that "looks like" palette you present.



THE INPUT

#8E938D

#93A39C

#95B8AF

#9FD1BE

#BFCDBD

#A9D1A7

#E6E496

	red	green	blue
[1,]	142	147	141
[2,]	147	163	156
[3,]	149	184	175
[4,]	159	209	190
[5,]	191	205	189
[6,]	169	209	167
[7,]	230	228	150

NO TRIVIAL TRANSFORMATION

Interpolation to fit max length would leads to trouble with palettes of the same scale but different lengths, and qualitative palettes.

Padding with zeroes would not work with diverging palettes.

HAND CRAFTING FEATURES

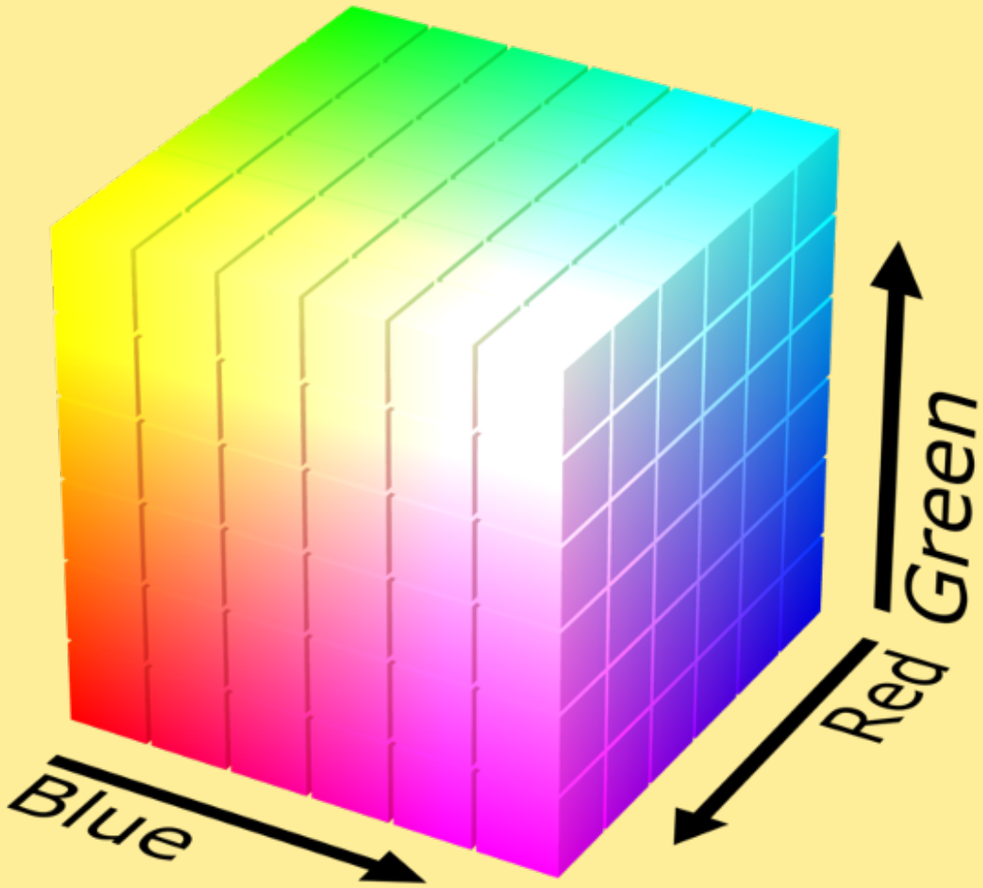
- similar to early image classification

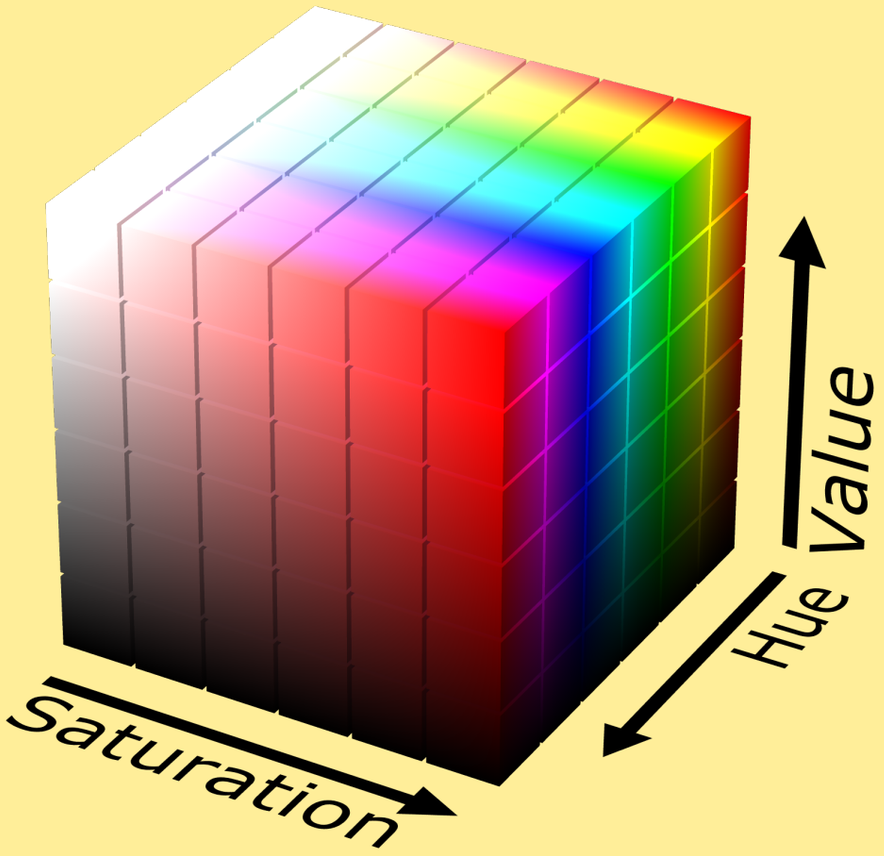
FEATURE LIST

- Includes a certain color
- All a certain color
- Length
- Type (sequential, divergent, qualitative. hardcoded)
- Linearity
- Twice linear (to identify divergent palettes)
- Colorblindness
- Minimal distance between points
- Maximal distance between points

Continuous palettes are taken with 20 points

COLORSPACES





Also CIELAB color space

METHOD

METHOD

- Create a tibble with a list column of the colors

METHOD

- Create a tibble with a list column of the colors
- Define a function for each feature

METHOD

- Create a tibble with a list column of the colors
- Define a function for each feature
- Use mutate() with each function

METHOD

- Create a tibble with a list column of the colors
- Define a function for each feature
- Use mutate() with each function
- Model

METHOD

- Create a tibble with a list column of the colors
- Define a function for each feature
- Use mutate() with each function
- Model
- …

METHOD

- Create a tibble with a list column of the colors
- Define a function for each feature
- Use mutate() with each function
- Model
- …
- Profit?

CREATING TIBBLE

R/data_list.R

```
##    package     palette        type length group      colors
## 1 ggthemes         Red sequential     20     c <chr [20]>
## 2 ggthemes       Green sequential     20     c <chr [20]>
## 3 ggthemes        Blue sequential     20     c <chr [20]>
## 4 ggthemes      Orange sequential     20     c <chr [20]>
## 5 ggthemes        Gray sequential     20     c <chr [20]>
## 6 ggthemes Blue Light sequential     20     c <chr [20]>
```

934 rows in total

FEATURE FUNCTIONS

```
color_contains <- function(pal, color, space) {
  spectrum1 <- convert_colour(t(col2rgb(pal)), "rgb", space)
  spectrum2 <- convert_colour(t(col2rgb(color)), "rgb", space)

  compare_colour(spectrum1, spectrum2, space, method = "cie2000") %>%
    min()
}
```

convert_colour() and compare_colour() from **farver** package

FEATURE FUNCTIONS

```
linear <- function(pal, space) {
  colors <- convert_colour(t(col2rgb(pal)), "rgb", space) %>%
    as.data.frame() %>%
    mutate(x = row_number())

  out <- lm(x ~ ., colors) %>%
    summary() %>%
    .$adj.r.squared

  if(is.nan(out))
    return(0)

  out
}
```

convert_colour() and compare_colour() from **farver** package

```r
data_num <- data_list %>%
  mutate(# contains all a certain color
         contains_yellow = map_dbl(colors, ~ color_contains(.x, "yellow", "hsv")),
         contains_orange = map_dbl(colors, ~ color_contains(.x, "orange", "hsv")),
         contains_red = map_dbl(colors, ~ color_contains(.x, "red", "hsv")),
         contains_purple = map_dbl(colors, ~ color_contains(.x, "purple", "hsv")),
         contains_blue = map_dbl(colors, ~ color_contains(.x, "blue", "hsv")),
         contains_green = map_dbl(colors, ~ color_contains(.x, "green", "hsv")),
         contains_brown = map_dbl(colors, ~ color_contains(.x, "brown", "hsv")),
         contains_white = map_dbl(colors, ~ color_contains(.x, "white", "hsv")),
         contains_black = map_dbl(colors, ~ color_contains(.x, "black", "hsv")),
         # All a certain color
         all_contains_yellow = map_dbl(colors, ~ color_all_contains(.x, "yellow", "hsv")),
         all_contains_orange = map_dbl(colors, ~ color_all_contains(.x, "orange", "hsv")),
         all_contains_red = map_dbl(colors, ~ color_all_contains(.x, "red", "hsv")),
         all_contains_purple = map_dbl(colors, ~ color_all_contains(.x, "purple", "hsv")),
         all_contains_blue = map_dbl(colors, ~ color_all_contains(.x, "blue", "hsv")),
         all_contains_green = map_dbl(colors, ~ color_all_contains(.x, "green", "hsv")),
         all_contains_brown = map_dbl(colors, ~ color_all_contains(.x, "brown", "hsv")),
         all_contains_white = map_dbl(colors, ~ color_all_contains(.x, "white", "hsv")),
         all_contains_black = map_dbl(colors, ~ color_all_contains(.x, "black", "hsv")),
         # Is it linear in a perceptually uniform space
         linear = map_dbl(colors, ~ linear(.x, "hunterlab")),
         linear_deutan = map_dbl(colors, ~ linear(deutan(.x), "hunterlab")),
         linear_protan = map_dbl(colors, ~ linear(protan(.x), "hunterlab")),
         linear_tritan = map_dbl(colors, ~ linear(tritan(.x), "hunterlab")),
         # Twice linear
         twice_linear = map_dbl(colors, ~ linear_split(.x, "hunterlab")),
         twice_linear_deutan = map_dbl(colors, ~ linear_split(deutan(.x), "hunterlab")),
         twice_linear_protan = map_dbl(colors, ~ linear_split(protan(.x), "hunterlab")),
         twice_linear_tritan = map_dbl(colors, ~ linear_split(tritan(.x), "hunterlab")),
```

```
  # Min distance between points
  min_distance = map_dbl(colors, ~ min_distance(.x, "hunterlab")),
  min_distance_deutan = map_dbl(colors, ~ min_distance(deutan(.x), "hunterlab")),
  min_distance_protan = map_dbl(colors, ~ min_distance(protan(.x), "hunterlab")),
  min_distance_tritan = map_dbl(colors, ~ min_distance(tritan(.x), "hunterlab")),
  # Max distance between points
  max_distance = map_dbl(colors, ~ max_distance(.x, "hunterlab")),
  max_distance_deutan = map_dbl(colors, ~ max_distance(deutan(.x), "hunterlab")),
  max_distance_protan = map_dbl(colors, ~ max_distance(protan(.x), "hunterlab")),
  max_distance_tritan = map_dbl(colors, ~ max_distance(tritan(.x), "hunterlab")),
  # IQR distance between points
  iqr_distance = map_dbl(colors, ~ iqr_distance(.x, "hunterlab")),
  iqr_distance_deutan = map_dbl(colors, ~ iqr_distance(deutan(.x), "hunterlab")),
  iqr_distance_protan = map_dbl(colors, ~ iqr_distance(protan(.x), "hunterlab")),
  iqr_distance_tritan = map_dbl(colors, ~ iqr_distance(tritan(.x), "hunterlab"))
  )
```

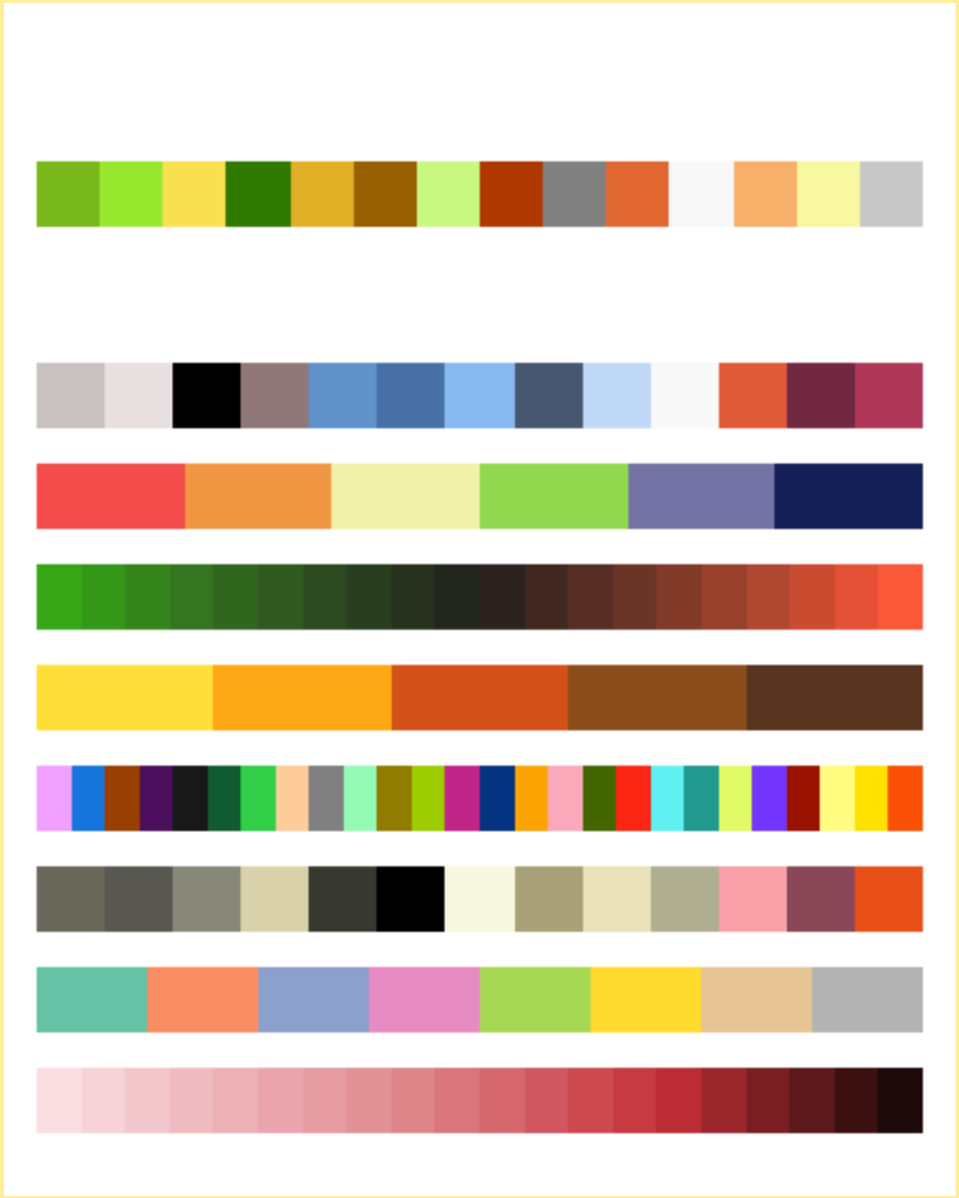
SCALING

```
data_meta <- data_num %>%
  mutate(id = as.character(str_glue("{package}-{palette}-{group}")))

data_prescaled <- data_meta %>%
  select(-package, -palette, -colors) %>%
  mutate_at(vars(type, group), funs(as.factor))

library(recipes)

re <- recipe(id ~ ., data = data_prescaled) %>%
  step_dummy(type, group) %>%
  step_center(all_predictors()) %>%
  step_scale(all_predictors()) %>%
  prep()

data_scaled <- bake(re, newdata = data_prescaled)
```

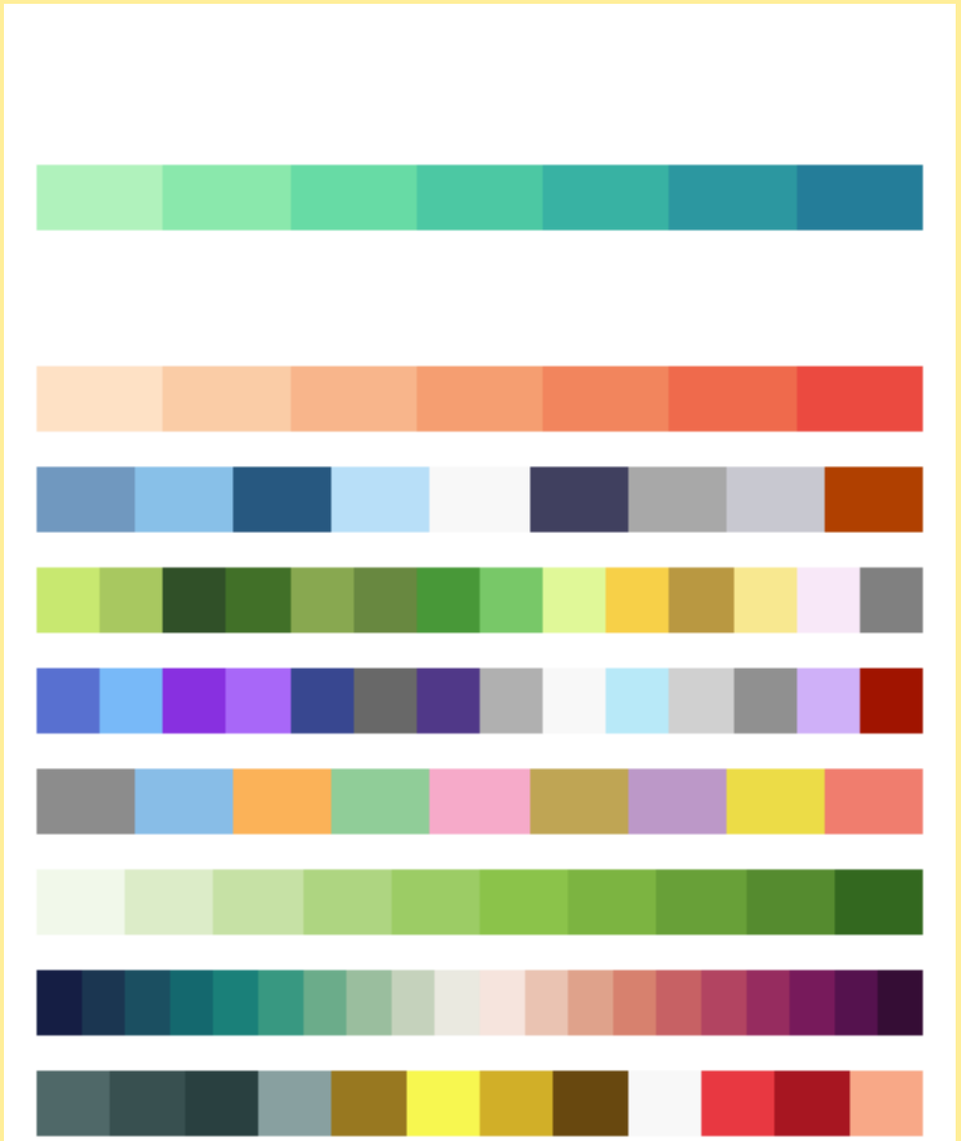
MODELING

- Simple nearest neighbors method
- Using Euclidean distance
- Same weights for all predictors due to scaling

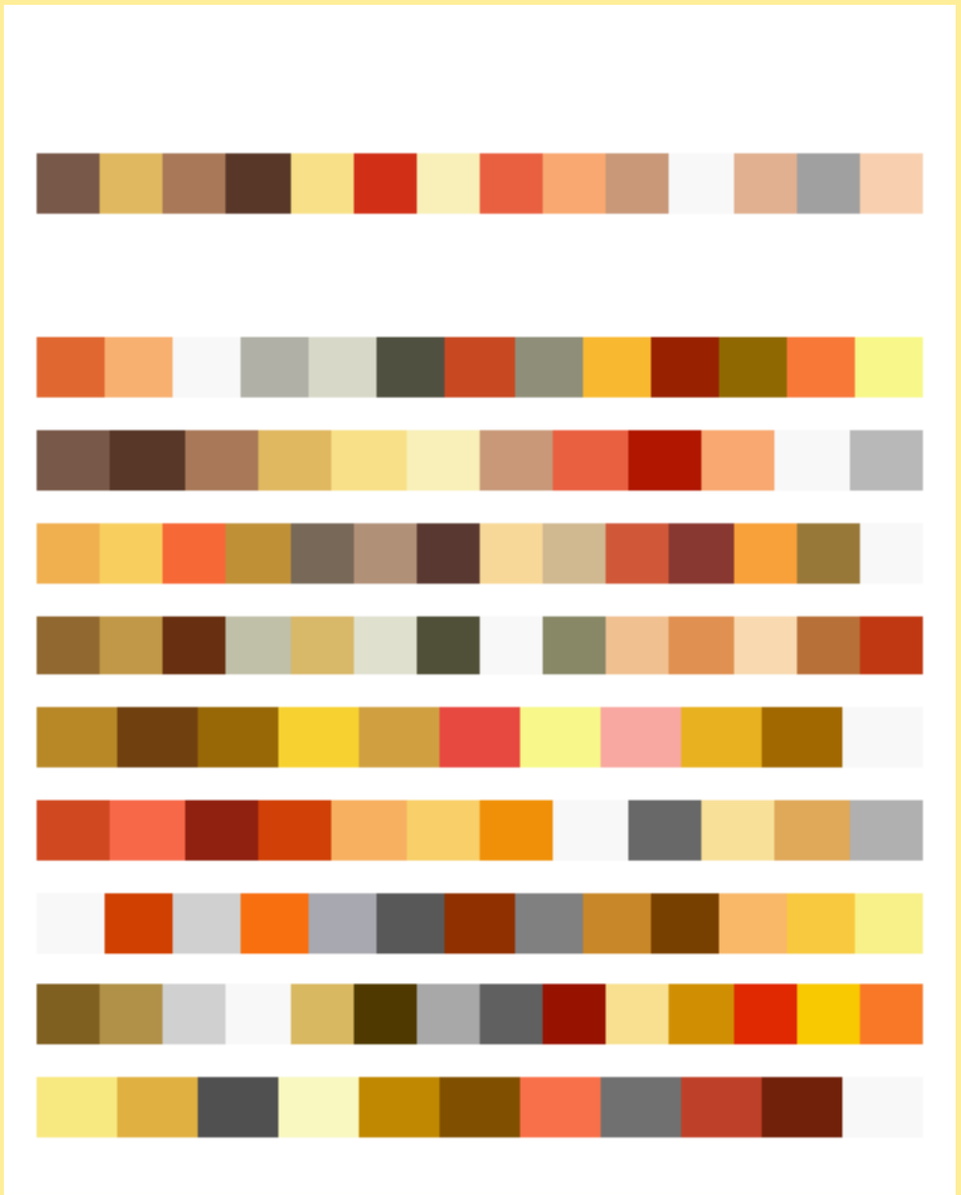
BASELINE

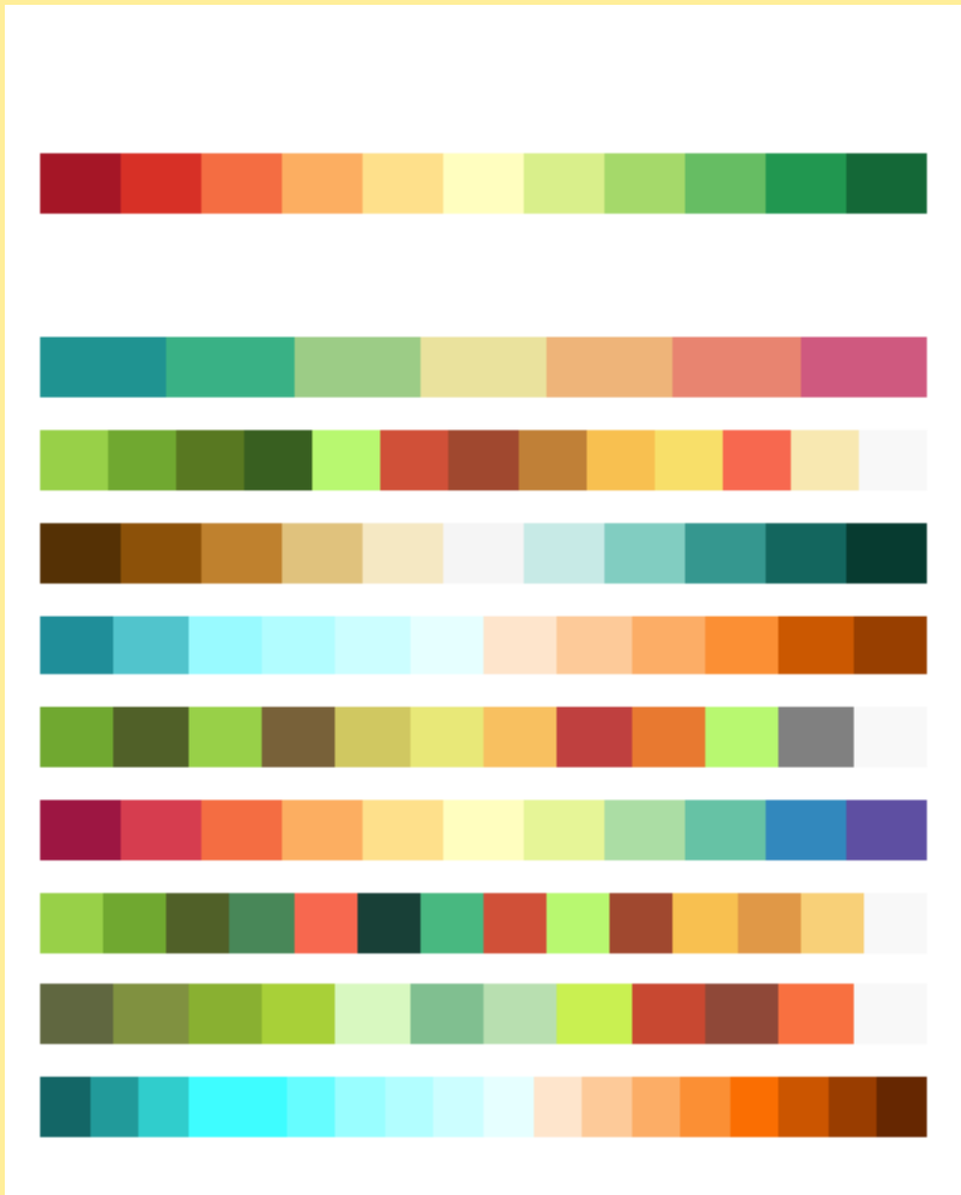


BASELINE

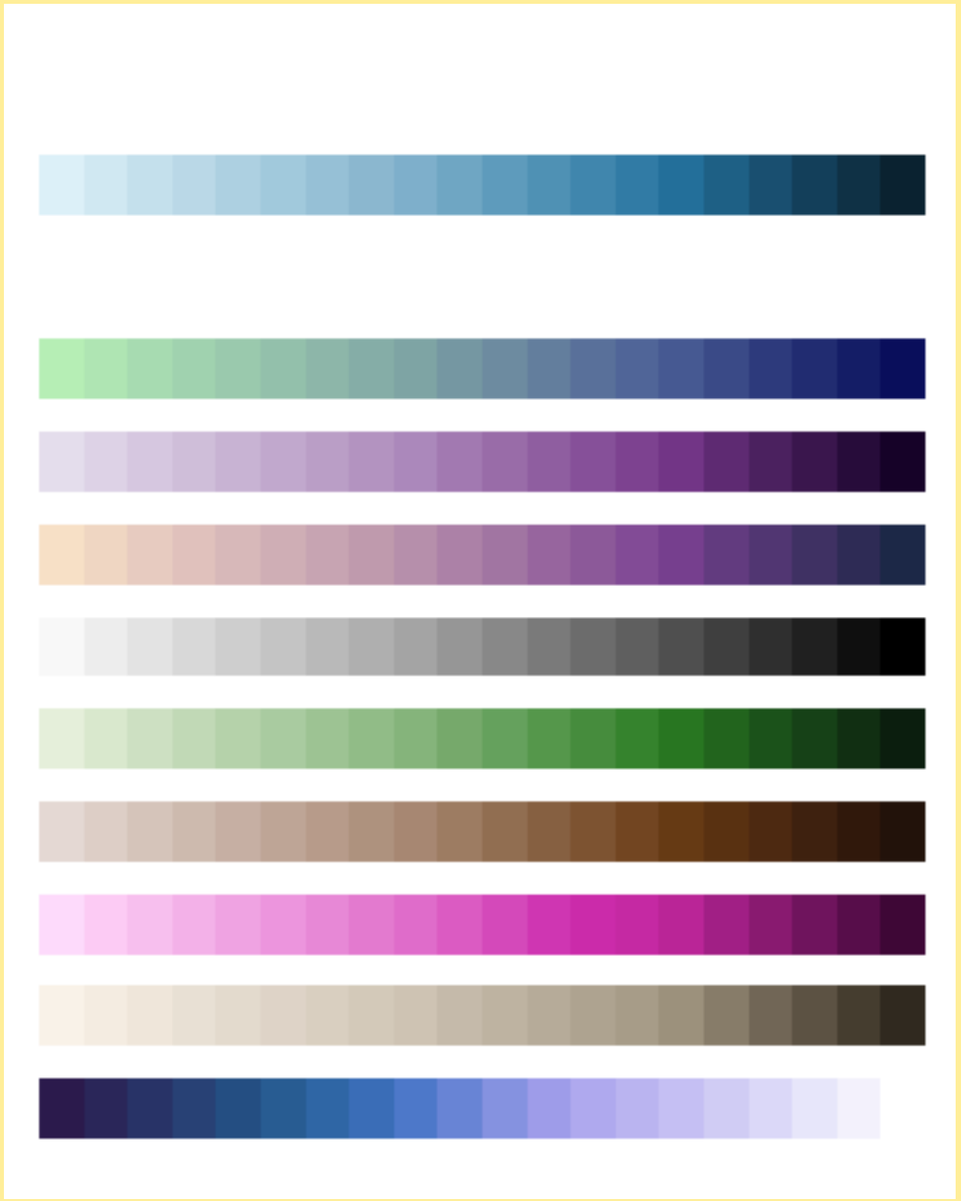


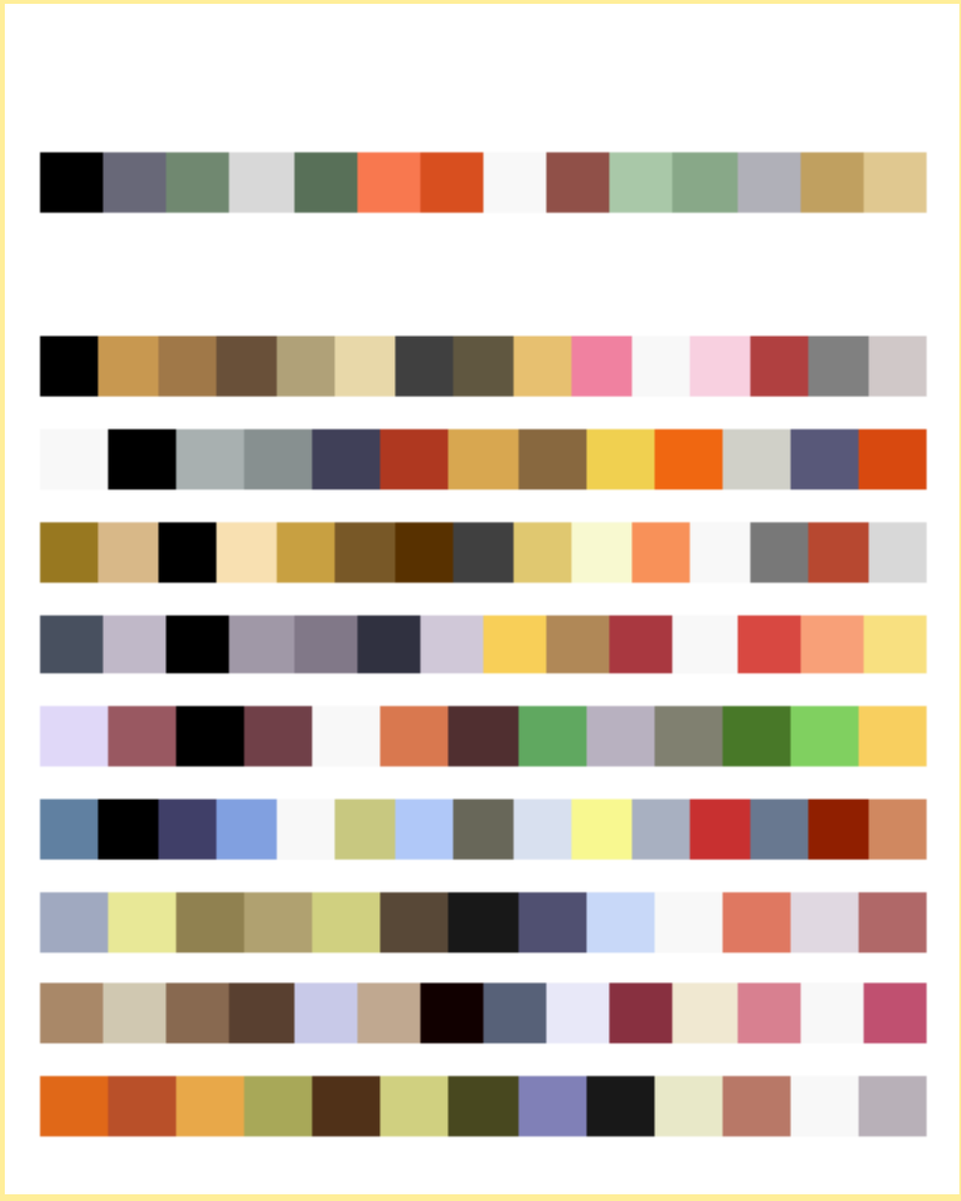
NEAREST NEIGHBORS





NEAREST NEIGHBORS





LIVE DEMO

With audience participation